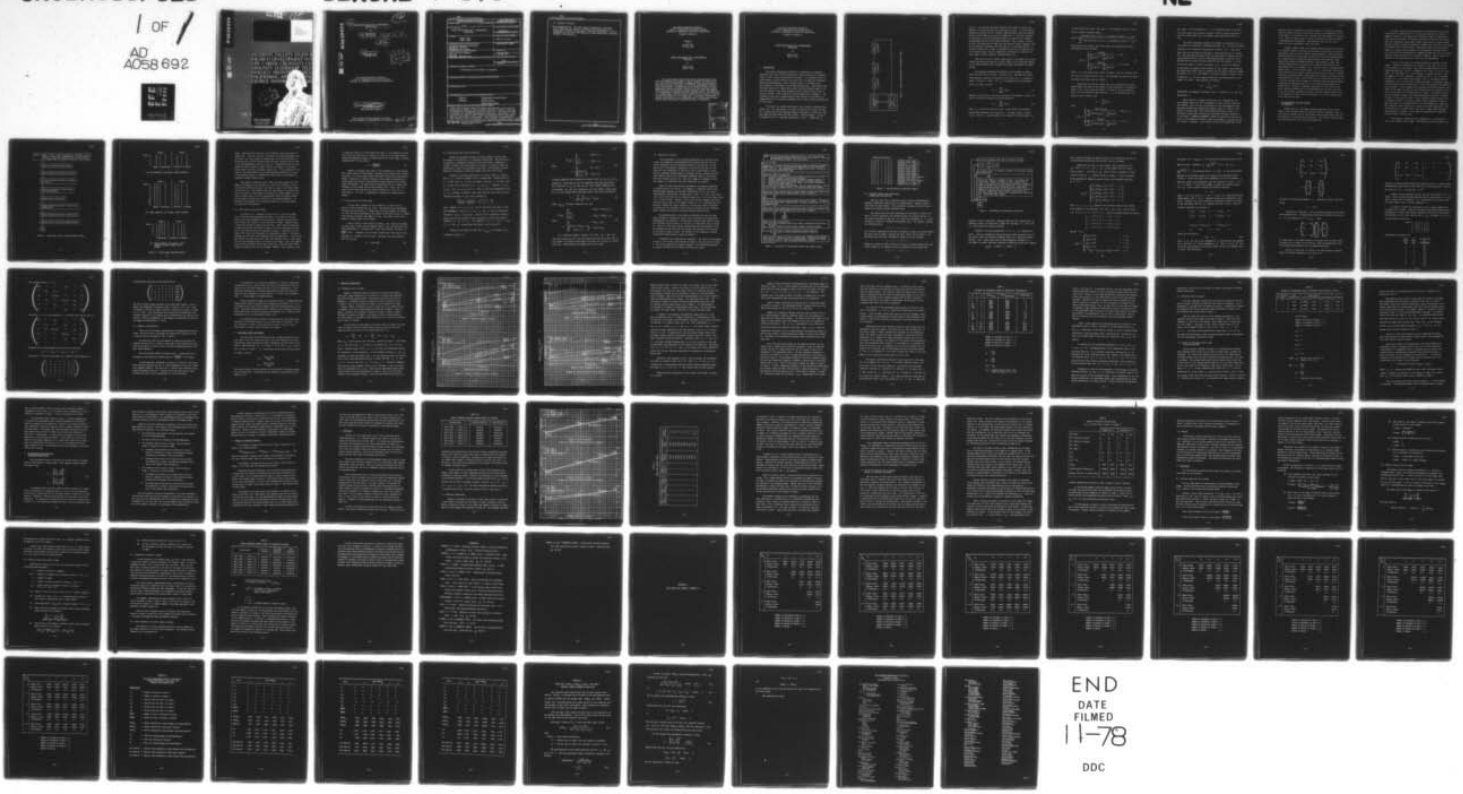
AD-A058 692 GEORGE WASHINGTON UNIV WASHINGTON D C PROGRAM IN LOG--ETC F/G 15/5
SPARES PROVISIONING FOR A HETEROGENEOUS POPULATION,(U)
JUN 78 D GROSS, J F INCE N00014-75-C-0729
UNCLASSIFIED SERIAL-T-376 NL

1 OF 1
AD A058 692



END
DATE FILMED
11-78
DDC

LEVEL II 12

THE GEORGE WASHINGTON UNIVERSITY

STUDENTS FACULTY STUDY R ESEARCH DEVELOPMENT FUT URE CAREER CREATIVITY CO MMUNITY LEADERSHIP TECH NOLOGY FRONTIE ESIGN ENGINEERING APP ENC GEORGE WASHIN NIV

INSTITUTE FOR MANAGEMENT SCIENCE AND ENGINEERING

SCHOOL OF ENGINEERING AND APPLIED SCIENCE

(12)

(6) SPARES PROVISIONING FOR A HETEROGENEOUS POPULATION,

by

(10) Donald/Gross
John F./Ince

(11) 16 Jun 78

(12) 78p.

(14) Serial-T-376
~~16 June 1978~~

The George Washington University
School of Engineering and Applied Science
Institute for Management Science and Engineering

(15) Program in Logistics
~~Contract~~ N00014-75-C-0729
Project NR 347 020
Office of Naval Research

405 337

LB

NONE

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER T-376	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* SPARES PROVISIONING FOR A HETEROGENEOUS POPULATION		5. TYPE OF REPORT & PERIOD COVERED SCIENTIFIC 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) DONALD GROSS JOHN F. INCE		8. CONTRACT OR GRANT NUMBER(s) N00014-75-C-0729
9. PERFORMING ORGANIZATION NAME AND ADDRESS THE GEORGE WASHINGTON UNIVERSITY PROGRAM IN LOGISTICS WASHINGTON, D. C. 20037		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS OFFICE OF NAVAL RESEARCH CODE 430D ARLINGTON, VIRGINIA 22217		12. REPORT DATE 16 June 1978 13. NUMBER OF PAGES 73
14. MONITORING AGENCY NAME & ADDRESS*(if different from Controlling Office)*		15. SECURITY CLASS. *(of this report)* NONE 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

DISTRIBUTION OF THIS REPORT IS UNLIMITED.

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*

QUEUEING
INVENTORY
HETEROGENEOUS MACHINE REPAIR
MACHINE REPAIR
SPARES PROVISIONING

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

The classic machine repair with spares queueing model can be used for determining the number of repair channels and spares inventory level when the population is homogeneous (all units identical). This paper develops a procedure for treating nonhomogeneous populations, specifically, a population with two types of items, each with exponential failure and repair times but with different mean values. Exact solutions are obtained for small population sizes and compared to an approximate procedure using (Cont'd)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
S/N 0102-014-6601 |

NONE

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

20. Abstract continued

the classical theory. The exact model is essentially a two-stage cyclic queue with two classes of customers and the extension to N stages and M customers is described. A priority discipline is also incorporated into the model and extension to other disciplines, including blocking, is discussed.

THE GEORGE WASHINGTON UNIVERSITY
School of Engineering and Applied Science
Institute for Management Science and Engineering
Program in Logistics

Abstract
of
Serial T-376
16 June 1978

SPARES PROVISIONING FOR A HETEROGENEOUS POPULATION

by

Donald Gross
John F. Ince

The classic machine repair with spares queueing model can be used for determining the number of repair channels and spares inventory level when the population is homogeneous (all units identical). This paper develops a procedure for treating nonhomogeneous populations, specifically, a population with two types of items, each with exponential failure and repair times but with different mean values. Exact solutions are obtained for small population sizes and compared to an approximate procedure using the classical theory. The exact model is essentially a two-stage cyclic queue with two classes of customers and the extension to N stages and M customers is described. A priority discipline is also incorporated into the model and an extension to other disciplines, including blocking, is discussed.

THE GEORGE WASHINGTON UNIVERSITY
School of Engineering and Applied Science
Institute for Management Science and Engineering
Program in Logistics

SPARES PROVISIONING FOR A HETEROGENEOUS POPULATION

by

Donald Gross
John F. Ince

1. Introduction

The spares provisioning problem for a population of completely repairable units can often be successfully treated using cyclic queueing theory. Such a system is described as follows. It is desired to have a given number of units (which we will often refer to simply as "machines") operating simultaneously. When a machine fails, it is replaced by a spare machine, if available, from a spare machines pool. The machine that has failed goes into a repair phase (which may have several stages to it) and then returns to the spare machines pool for eventual reuse; thus, a cycle is created. Such a system is shown schematically in Figure 1. The probability that when a machine fails a spare machine is available is a common measure of the system performance and is called "spares availability" or "fill rate."

If failures are Poisson and service times (removal, transportation, repair, etc.) are exponential, it is not too difficult to obtain the number of spares and service channels needed to guarantee a certain fill rate [see GROSS and INCE (1978) and INCE (1978)]. Viewing Figure 1, we see it is desired to have M machines operational so that y spares and c_i

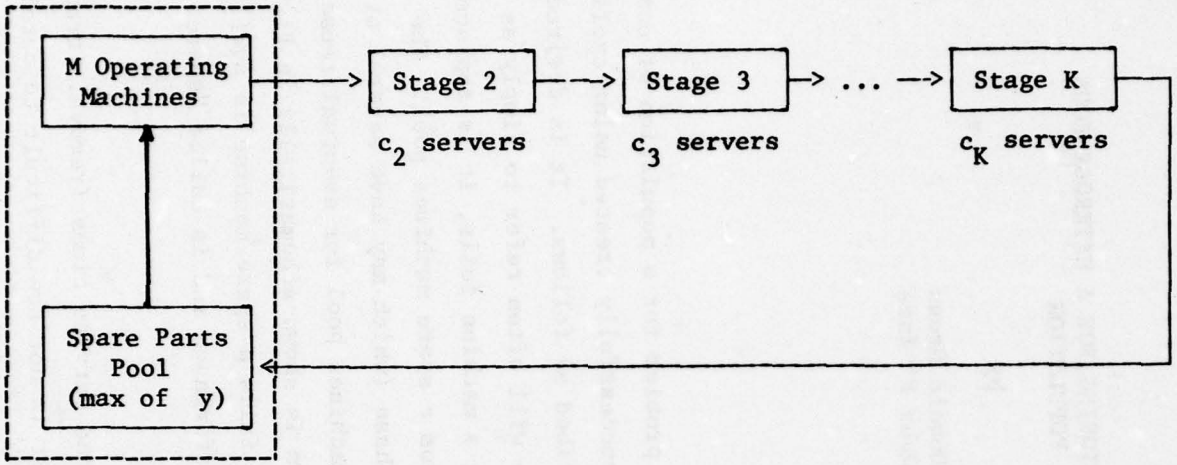


Figure 1. Spares provisioning for repairable items.

(i=2,3,...,k) servers (decision variables) are needed to support the system at a desired service level (say 90% fill rate). If the operating machines and spares pool are "lumped" together (dotted line) and considered the first stage in the cycle (a stage with M "service" channels), we then have a true cyclic queueing system, the queue size at stage 1 representing the size of the spares inventory. When a stage 1 "server" is idle, the spares pool is empty and the system is short a machine; that is, the system is operating with less than the M machines desired. A "service completion" at stage 1 represents the failure of a machine and at any other stage a "service completion" represents the completion of that stage's function (removal, transportation, etc.).

When there is only a single repair phase, a two-stage cyclic queue results (an operating stage and a repair stage), and the problem reduces to the classic machine repair problem with spares [see GROSS and HARRIS (1974)].

The marginal probability distribution of the number at stage 1 yields the fill rate as follows. Letting $p_1(n)$ represent the steady state marginal probability distribution that there are n customers at stage 1 (n=0,1,2,...,M+y), and F_v the probability that the spares pool is not empty, we have

$$F_v = \sum_{n=M+1}^{M+y} p_1(n) . \qquad (1)$$

However, fill rate has to do with the state of the spares pool upon a request (a failure) and hence the desired measure is

$$F_A = \sum_{n=M+1}^{M+y} q_1(n) , \qquad (2)$$

where F_A is the fill rate and $q_1(n)$ is the steady state marginal *conditional* probability that there are n at stage 1 *when* a failure occurs. Hence F_v can be viewed as the "virtual" fill rate, while F_A

is the "actual" fill rate. The $q_1(n)$ can be obtained from the $p_1(n)$ by use of Bayes' theorem; namely,

$$q_1(n) = \frac{\Pr\{\text{failure about to occur} \mid n \text{ customers at stage 1}\} \cdot p_1(n)}{\sum_{n=0}^{M+y} \Pr\{\text{failure about to occur} \mid n \text{ customers at stage 1}\} \cdot p_1(n)} .$$

When failures are Poisson and service times are exponential we have a birth-death process yielding

$$q_1(n) = \begin{cases} \dfrac{n\mu_1 p_1(n)}{\sum_{n=0}^{M} n\mu_1 p_1(n) + \sum_{n=M+1}^{M+y} M\mu_1 p_1(n)}, & n \leq M \\ \dfrac{M\mu_1 p_1(n)}{\sum_{n=0}^{M} n\mu_1 p_1(n) + \sum_{n=M+1}^{M+y} M\mu_1 p_1(n)}, & M < n \leq M+y \end{cases} \tag{3}$$

where μ_1 is the "mean service rate" at stage 1; that is, the mean failure rate of a machine and the $p_1(n)$ are obtainable via the cyclic queueing theory [see Gross and Ince (1978)].

When there are only two stages (the classic machine repair with spares model) the state of the system is often measured by the number at stage 2 (i.e., in or awaiting repair). If this is the case, denoting $q_2(n)$ as the probability that there are n in or awaiting repair (in stage 2) upon a failure, then

$$F_A = \sum_{n=0}^{y-1} q_2(n) ,$$

where

$$q_2(n) = \begin{cases} \dfrac{(M+y-n)\mu_1 p_2(n)}{\sum_{n=y+1}^{M+y} (M+y-n)\mu_1 p_2(n) + \sum_{n=0}^{y} M\mu_1 p_2(n)}, & M+y \geq n \geq y \\ \dfrac{M\mu_1 p_2(n)}{\sum_{n=y+1}^{M+y} (M+y-n)\mu_1 p_2(n) + \sum_{n=0}^{y} M\mu_1 p_2(n)}, & 0 \leq n < y \end{cases}$$

and $p_2(n)$ is the probability of n at or awaiting repair at any arbitrary time point and is easily calculable from the classical machine repair with spares model [see, for example, Gross and Harris (1974), p. 123].

The above development assumes all machines are identical; that is, each machine has exponential lifetimes with mean $1/\mu_1$ and can be repaired according to an exponential distribution with mean $1/\mu_2$. This paper deals with the case where all machines in the population are not identical. This realistically often occurs in populations of complex, new generation machines where there is technological growth so that the newer machines may have better failure characteristics (say smaller μ_1) and may also be easier to repair (say larger μ_2). For ease of analysis, in the sequel we change notation slightly so that stage 1 (the operating and spares pool stage) looks in notation like the repair stage (or stages). We replace M by c_1 (the number of "service" channels in stage 1) and we denote by N the total number of machines (including spares) in the system ($N = M+y$). Hence Equation (2) becomes

$$F_A = \sum_{n=c_1+1}^{N} q_1(n) , \qquad (4)$$

and Equation (3) changes accordingly, with M replaced by c_1 and $M+y$ replaced by N.

While there is a sizable literature on the "homogeneous" case [all machines alike; see the bibliography in Gross and Ince (1978) and Ince (1978)], there are very few references on the heterogeneous case. The major problem when considering different types or "classes" of machines in a first come, first served (FCFS) discipline is that it is necessary to keep track of where each individual customer (machine) is in the system. POSNER and BERNHOLTZ (1968b) extended their previous two works on the homogeneous problem, (1967) and (1968a), to consider different classes of machines. They avoided the task of keeping track of the

specific order of customers of different classes in a queue by assuming that the service rate for a particular class was state-dependent (on the total number of customers of that class and the total number of customers of all classes at that stage) and could be obtained by using the mean of a hypergeometric distribution. Unfortunately, this solution can be shown to be only approximate rather than exact.

BASKETT, CHANDY, MUNTZ, and PALACIOS (1975) addressed the problem of different classes of customers and obtained solutions for four service disciplines associated with computer systems modeling: FCFS; processor sharing, where each customer gets a share of and is served simultaneously by a single server; no queueing (ample server); and last-come, first-served with preemptive/resume servicing. Unfortunately, stages with a FCFS discipline had to be restricted to having the same service time distributions for all customers, i.e., only one class of customers (homogeneous case). In other words, Baskett, *et al.*, found that a FCFS discipline with finite (vs. ample) servers and more than one class of customers could not be handled without considering the ordering of customers in the specification of a state. KELLY (1975) set up a notational structure for a network of queues with customers of different types and proved theorems concerning the uniqueness and form of the equilibrium distribution of system states. However, a numerical application of the results would still involve the generation of and accounting for the vast state space of all the possible orderings in queues, which has always been the crux of the problem if actual results are desired. It is this problem which is of chief concern here.

2. Two-stage Model with Two Classes of Customers

Treated here is a two-stage (operating and repair--the classic machine repair problem) model but with two classes of machines; that is, each class of machines has its own failure rate and mean repair time, but all machines in a given class are identical. The failure time and repair time *distributions* of each class are exponential.

We first develop an exact procedure for generating the states and solving for the probabilities for a two-stage system with an arbitrary number of servers at each stage and an arbitrary number of customers of each type in the system. The service discipline is FCFS and all service times are exponential unless ample servers are present at a stage, whereupon the results of Posner and Bernholtz (1968a) can be invoked, which allows for general service distributions. Approximate models are also defined (based on a single class of machines--the homogeneous classical model--with characteristics determined by a weighted average over the two classes), numerical results obtained, and the models compared. Figure 2 gives an overview of the algorithm used to carry out the computations.

A state of the system is a unique ordering of customers at *both* stages. Consider a system with one customer of each type, for a total of two customers in the system. Since both customers could be at one stage simultaneously, there is a maximum of two "slots" in front of each stage. Let the number 1 in a slot represent a customer of type 1, and the number 2 in a slot represent a customer of type 2. Let 0 represent an empty slot. For single server stages, that is, one machine desired operational (one is a spare) and a single repairman, the possible states of the system are shown in Figure 3(a). The solid vertical line separates the two stages. The two dashed lines represent the division between customers in the queue (to the left of the dashed line) and customers in service (to the right of the dashed line) at a stage. For example, state 1 has no customers in stage 1 (no operating machine or spare), a customer of type 1 in the queue at stage 2 (type 1 machine awaiting repair), and a customer of type 2 in service at stage 2 (type 2 machine in repair). Note that there is no predisposition as to which machine is a spare and which operates; they alternate as a result of the FCFS discipline. One can visualize the customer flow as from left to right, with customers leaving service at stage 2 to the right and cycling back to enter the queue at stage 1 from the left.

Now consider a system with two customers of type 1 and one of type 2, with single server stages. Now there will be three "slots" per

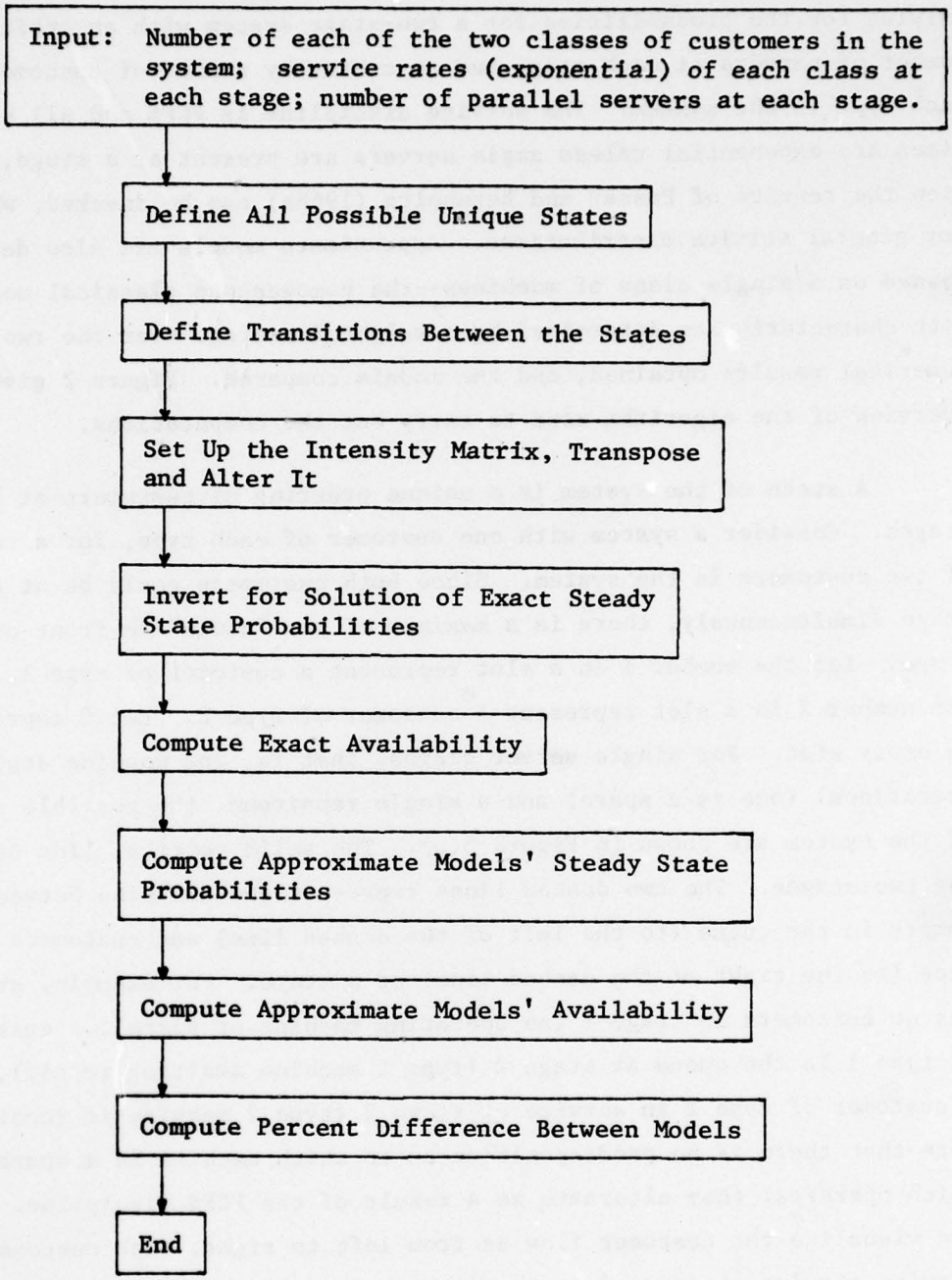


Figure 2. Algorithm of exact and approximate models.

	Stage 1		Stage 2	
State	Spares	Operating	Waiting	In Repair
1	0	0	1	2
2	0	0	2	1
3	0	1	0	2
4	0	2	0	1
5	1	2	0	0
6	2	1	0	0

(a) Two machines, two stages, single channels.

	Stage 1		Stage 2	
State	Spares	Operating	Waiting	In Repair
1	0 0	0	1 1	2
2	0 0	0	1 2	1
3	0 0	0	2 1	1
4	0 0	1	0 1	2
5	0 0	1	0 2	1
6	0 0	2	0 1	1
7	0 1	1	0 0	2
8	0 1	2	0 0	1
9	0 2	1	0 0	1
10	1 1	2	0 0	0
11	1 2	1	0 0	0
12	2 1	1	0 0	0

(b) Three machines, two stages, single channels.

	Stage 1		Stage 2	
State	Operating		Waiting	In Repair
1	0 0 0		1	1 2
2	0 0 0		2	1 1
3	0 0 1		0	1 2
4	0 0 2		0	1 1
5	0 1 1		0	0 2
6	0 1 2		0	0 1
7	1 1 2		0	0 0

(c) Three machines, two stages, three channels in first stage, two in second.

Figure 3. System stage characterization.

stage. Using the above notation, the 12 possible states are shown in Figure 3(b). Since the two type 1 customers are indistinguishable from each other, the total number of permutations is cut down from what it would have been if there had been three distinct customers (two of type 1, but distinguishable, and one of type 2). Assuming now that the above system has three servers at stage 1 and two at stage 2, the possible states reduce to seven, as shown in Figure 3(c). Since customers to the right of the dotted lines in each stage represent customers in service in the parallel servers, no additional *unique* states are generated by juggling customers among servers.

The number of possible unique states increases sharply with the number of customers in the system; e.g., with five customers of each type and five servers at stage 1 and three at stage 2, the total number of states is 524. This number would have been greater for smaller numbers of servers at either stage, and smaller for larger numbers of servers at either stage. We will now determine easily calculable bounds for the number of states as a function of the system parameters. In addition, an expression for determining numerically the exact number of states will be developed.

2.1 State space size--upper bound

Let there be N customers of type 1 and M of type 2, where type 1 items are distinguishable from type 2, but items of the same type are not distinguishable from each other. Consider any one ordering of these $N+M$ items in a line from left to right. Imagine taking a stick and placing it between any two items. That could now represent a unique state where the number of items to the left represents the number of each type and order of customers at stage 1. Similarly, to the right of the stick could represent the situation at stage 2. Any other placement of the stick could represent another unique state. If the stick were placed at either end of the $N+M$ long customer line, it could represent all the customers being in one stage or the other. Now, just consider the stick as another item to put in the line. There are thus $N+M+1$ things to arrange, which can be arranged $(N+M+1)!$ ways. However, since

N items are identical to each other, and since M are identical to each other, and since the stick is identical to itself, by the usual combinations and permutations logic, an upper bound on the total number of unique orderings (call $\overline{S}$) is given by

$$\overline{S} = \frac{(N+M+1)!}{N!\ M!\ 1!} . \qquad (5)$$

However, in general this limit is not achievable in a real system. With single servers at both stages, customers cannot change their relative positions, i.e., they cannot "pass" each other as they can at multiserver stages. Therefore, an initial ordering in the system is preserved in a cyclic manner and not all of the possible states predicted by the upper bound are feasible. The upper limit is achievable for up to three total customers in the system, with any mix between types 1 and 2, but for a case of two of each type with an initial condition of $1|2 \mid 1|2$, a state such as $1|1 \mid 2|2$ is impossible to achieve with single servers present.

2.2 State space size--lower bound

In this case, consider all the customers at a stage to be in service so that ordering does not count. The problem now is determining how many ways one can put N things into two boxes (stages), any number to a box, and then multiplying by the number of ways one can put M things into the same two boxes, any number to a box. The result is the total number of unique states of the system.

Using a similar "stick theory" as before, N+1 items (N customers of type 1 plus 1 stick) can be arranged (N+1)! ways, but because of the indistinguishable N customers and 1 stick (with itself), the result is $\frac{(N+1)!}{N!1!} = N+1$. Similarly, for the type 2 items we get M+1 ways of distributing M customers into two boxes. Therefore, the lower bound is given by (call $\underline{S}$)

$$\underline{S} = (N+1)(M+1) . \qquad (6)$$

2.3 State space size--exact calculation

Bounds are convenient because the actual number of states does not appear to be reducible to a convenient closed form. The upper bound is basically a permutation problem and the lower bound a combination problem. However, the actual number of states is a little of both, and essentially a brute force approach is now taken. The problem is decomposed to a permutation problem and summed to get the final result.

Consider a two-stage system with N of type 1 and M of type 2 with c_1, c_2 servers at stages 1 and 2, respectively. Assume there are n_1 of type 1 and m_1 of type 2 at stage 1, and that i_1 of type 1 and j_1 of type 2 are in service (i.e., $i_1 + j_1 \leq c_1$). Clearly, $i_1 \leq n_1$, $j_1 \leq m_1$, and $i_1, j_1, n_1, m_1 \geq 0$. There is only one arrangement of the $i_1 + j_1$ customers in service, since order does not count there, but the remaining customers, if any, can be arranged in

$$\frac{\{(n_1-i_1 + m_1-j_1)\}!}{(n_1-i_1)!\ (m_1-j_1)!} = \binom{n_1 + m_1 - c_1}{n_1 - i_1}$$

ways. However, if $c_1 \geq n_1 + m_1$, all are in service and there is only one arrangement. Also, if $n_1 + m_1 > c_1$, the i_1 must be restricted because if $(n_1-i_1) > (n_1+m_1-c_1)$, then we have an impossible condition where there are not enough of type 2 available to fill up the service channel, i.e., $m_1 < c_1 - i_1$. For example, if $n_1 = 2$, $m_1 = 3$, and $c_1 = 4$, then i_1 can only take the values 1 and 2, and not 0.

Therefore, the number of ways (call S_{n_1,m_1}) to arrange n_1, m_1 customers at stage 1 is

$$S_{n_1,m_1} = \begin{cases} 1 & ; \quad n_1+m_1 \leq c_1 \\ \sum_{i_1=0}^{n_1} X & ; \quad n_1+m_1 > c_1 \end{cases}$$

where (7)

$$X = \begin{cases} 0 & ; \quad m_1 < c_1-i_1 \\ \binom{n_1+m_1-c_1}{n_1-i_1} & ; \quad m_1 \geq c_1-i_1 \end{cases}$$

A similar summation can be derived for stage 2 with n_2, m_2, c_2, and i_2 parameters. The product of the two summations then gives the results for a given set of n_1, n_2, and m_1, m_2 ; and a double summation over the possible n's and m's (subject to the total number of each type, N, M , in the system) yields the final result, which can be written as

$$S = \sum_{m_1=0}^{M} \sum_{n_1=0}^{N} S_{n_1,m_1} \cdot S_{n_2,m_2} , \tag{8}$$

where S_{n_1,m_1} is given in Equation (7), and

$$S_{n_2,m_2} = \begin{cases} 1 & ; \quad (N-n_1) + (M-m_1) \leq c_2 \\ \sum_{i_2=0}^{N-n_1} Y & ; \quad (N-n_1) + (M-m_1) > c_2 \end{cases}$$

where (9)

$$Y = \begin{cases} 0 & ; \quad M-m_1 < c_2-i_2 \\ \binom{(N-n_1) + (M-m_1) - c_2}{(N-n_1) - i_2} & ; \quad M-m_1 \geq c_2-i_2 \end{cases}$$

As a numerical example, consider N=5 , M=2 , c_1=4 , and c_2=2 . The lower bound from (6) is 18; the upper bound from (5) is 168. The actual number of possible unique states, from (8), (9), and (7), is 76 for this case. Unfortunately, the bounds are not very tight.

2.4 Generation of states

The requirements of any algorithm generating the states are that it must be exhaustive in determining every possible case (so that all states communicate and the transition matrix is irreducible), it must keep track of the specific ordering in a state after the state is defined (so that transitions to and from it can be identified), it must weed out redundant states (caused by parallel servers, wherein distributions among servers do not distinguish different states), and it must save only feasible states (i.e., the total numbers of customers of the two classes in a generated state must agree with the inputs).

While the above are obvious requirements, creating an algorithm to satisfy them is not trivial. For example, a completely brute force approach might generate all possible combinations of 0's, 1's, and 2's (representing, respectively, an empty slot, a type 1 in a slot, and a type 2 in a slot) and throw out the unfeasible ones (e.g., a 0 to the right of any 1 or 2 in the string). However, five of each type of customer (and therefore ten slots at each of two stages) results in 3^{20} orderings (some 3.5 billion), taxing even modern computers. On the other hand, an approach of putting all the customers in some initial ordering and then juggling them to produce all possible orderings requires tedious logic.

The approach used in this paper was basically to count in base-three arithmetic, leaving out zeroes, and to deal with the stages semi-independently [as opposed to a continuously ordered string from the first (or left-most) slot of stage 1 to the last (or right-most) slot of stage 2]. This is a "semi-brute force" approach that has the advantage of exhaustiveness but can also take into account the structure of the system to cut down on cases to check. On overview of the algorithm to define the feasible unique states is given in Figure 4.

An example will illustrate the algorithm. Let there be two machines of type 1, one of type 2, two servers at stage 1, and one server at stage 2. The states generated are shown in Figure 5; the algorithm generates 14 states, of which ten are saved as feasible and unique.

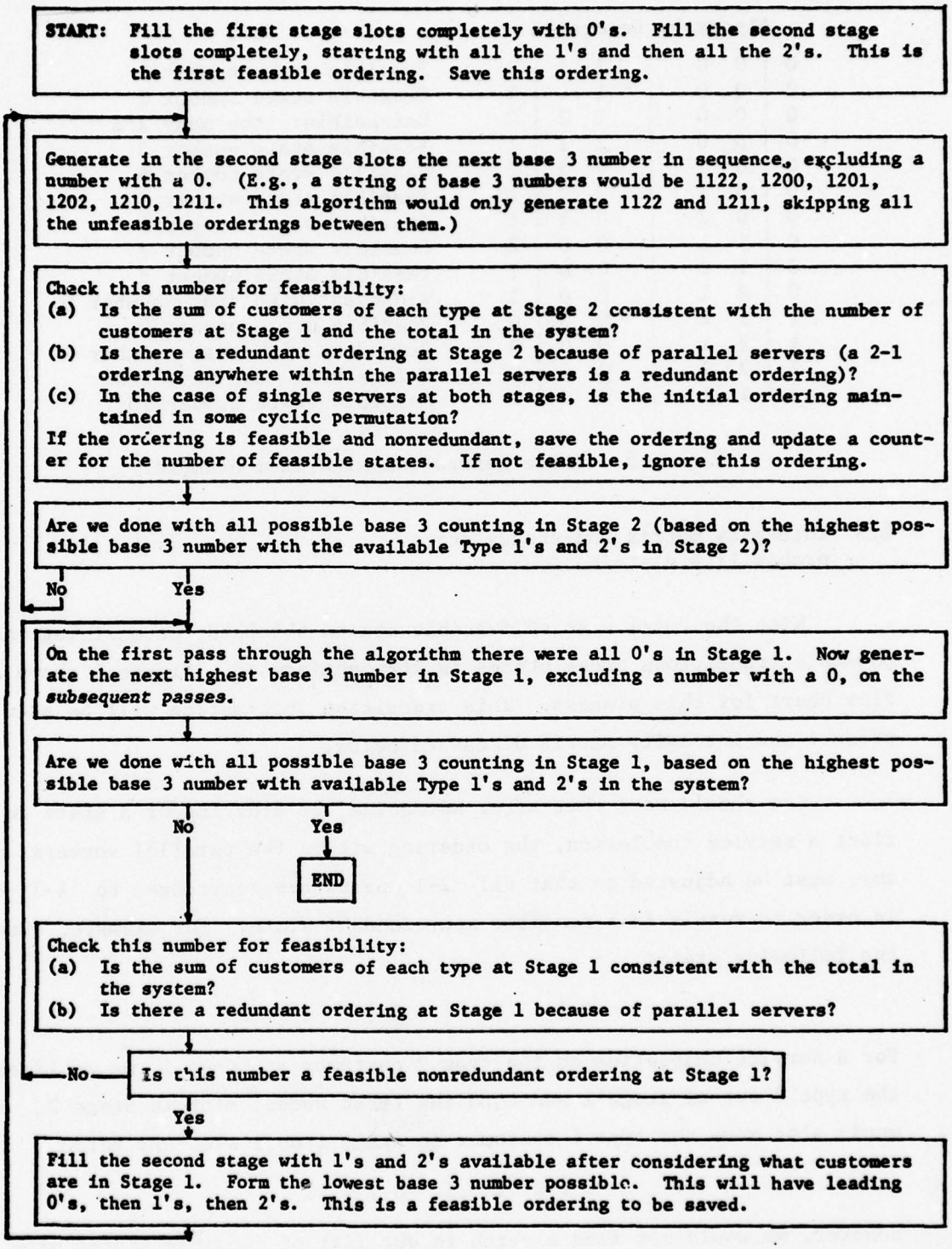


Figure 4. Algorithm for generating feasible and unique states.

Algorithm			Generated			Note
0	0	0	1	1	2	Feasible state number 1
0	0	0	1	2	1	Feasible state number 2
0	0	0	1	2	2	Unfeasible: too many 2's
0	0	0	2	1	1	Feasible state number 3
0	0	1	0	1	2	Feasible state number 4
0	0	1	0	2	1	Feasible state number 5
0	0	2	0	1	1	Feasible state number 6
0	1	1	0	0	2	Feasible state number 7
0	1	2	0	0	1	Feasible state number 8
0	2	1	0	0	1	Redundant with state number 8
1	1	2	0	0	0	Feasible state number 9
1	2	1	0	0	0	Redundant with state number 9
1	2	2	0	0	0	Unfeasible: too many 2's
2	1	1	0	0	0	Feasible state number 10

Figure 5. State generation algorithm example.

2.5 Intensity matrix and steady state probability distribution

With the total list of feasible states obtained, determining the possible transitions among states is straightforward. Figure 6 shows a flow chart for this process. This transition information will be used to produce the intensity matrix discussed below.

One should note that after advancing the ordering of a state to reflect a service completion, the ordering within the parallel servers, if any, must be adjusted so that all 2-1 pairs are rearranged to 1-2 pairs in order to result in a feasible nonredundant state. For example, consider the following state:

$$0 \mid 1\ 2 \quad \mid \quad 0 \mid 0\ 1\ .$$

For a service completion on the type 2 customer at stage 1, we would advance the type 2 out of stage 1 and into the first vacant slot in stage 2. We would also move the type 1 customer in stage 1 up a slot and get:

$$0 \mid 0\ 1 \quad \mid \quad 0 \mid 2\ 1\ .$$

However, we would not find a match in our list of feasible states with this result because it is redundant with the transition to the saved state:

$$0 \mid 0\ 1 \quad \mid \quad 0 \mid 1\ 2\ .$$

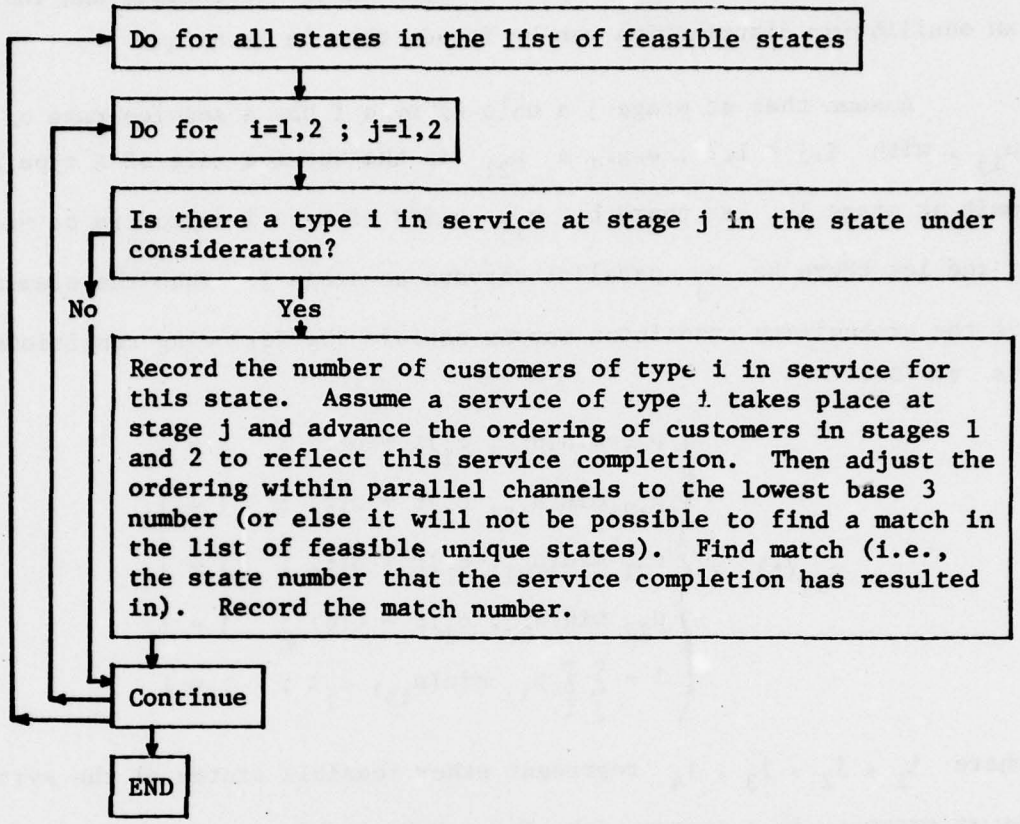


Figure 6. Algorithm for determining transitions.

Therefore, after a transition, ordering within parallel channels must be checked to yield the lowest base 3 number (by replacing 2-1 pairs with 1-2 pairs within the parallel servers).

Consider a two-stage system with N , M , c_1 , c_2 parameters as before. With the exponential servers we have a birth and death type process which, using the notation of BARLOW (1962), is a stationary Markov process, $X(t)$, whose state space is the nonnegative integers (i.e., the number of states) and whose transition probability matrix has elements:

$$p_{ij}(t) = p\{X(t) = j \mid X(0) = i\} .$$

Kelly (1975) has shown this Markov process to be irreducible and that if an equilibrium distribution can be found, then it is unique.

Assume that at stage j a unit of type i has a service rate of μ_{ij}, with $i,j = 1,2$, e.g., a μ_{21} is the service rate of a type 2 unit at stage 1. Let there be n_{ij} units of type i customers at stage j and let there be c_j parallel servers at stage j. Then the elements of the probability transition matrix satisfy the following conditions as $t \to 0$:

$$p_{ij}(t) = \begin{cases} \mu_{11} \min(n_{11}, c_1)t + o(t) ; & j = j_1 \\ \mu_{12} \min(n_{12}, c_2)t + o(t) ; & j = j_2 \\ \mu_{21} \min(n_{21}, c_1)t + o(t) ; & j = j_3 \\ \mu_{22} \min(n_{22}, c_2)t + o(t) ; & j = j_4 \\ 1 - \sum_j \sum_i \mu_{ij} \min(n_{ij}, c_j) ; & j = i \end{cases},$$

where j_1, j_2, j_3, j_4 represent other feasible states of the system. As an example, in a system with $N=3$, $M=3$, $c_1=3$, $c_2=3$, let the state i be 0 0 0 ¦ 1 1 2 | 0 0 0 ¦ 1 2 2. Then if we define states as follows,

j_1 : 0 0 0 ¦ 0 1 2 | 0 0 1 ¦ 1 2 2

j_2 : 0 0 1 ¦ 1 1 2 | 0 0 0 ¦ 0 2 2

j_3 : 0 0 0 ¦ 0 1 1 | 0 0 2 ¦ 1 2 2

j_4 : 0 0 2 ¦ 1 1 2 | 0 0 0 ¦ 0 1 2

then as $t \to 0$,

$$p_{ij}(t) = \begin{cases} 2\mu_{11}t + o(t) & ; \quad j = j_1 \\ \mu_{12}t + o(t) & ; \quad j = j_2 \\ \mu_{21}t + o(t) & ; \quad j = j_3 \\ 2\mu_{22}t + o(t) & ; \quad j = j_4 \\ 1 - (2\mu_{11} + \mu_{12} + \mu_{21} + 2\mu_{22})t + o(t) & ; \quad j = i \end{cases}.$$

The matrix $P(t) = \{p_{ij}(t)\}$ is the transition probability matrix of the Markov process. Defining $q_{ij} = \lim_{t\to 0} \frac{p_{ij}(t)}{t}$, $i \neq j$, and $q_{ii} = \lim_{t\to 0} \frac{p_{ii}(t) - 1}{t}$, the resulting matrix, $Q = \{q_{ij}\}$, is the infinitesimal generator of the Markov process, also known as the *intensity matrix*. Letting Π denote the steady state probability vector with elements π_i being the steady state probability that the system is in state i, $i=1, 2,\ldots,I$, where I *is the total number of unique and feasible states*, the π_i's can be found by solving the system of equations

$$0 = \pi Q ,$$

where 0 is a row vector of I 0's, Q is the $I\times I$ infinitesimal generator matrix, and π the I component steady state probability (row) vector. However, one of the equations is redundant and must be replaced by the requirement that the π_i elements must sum to unity. Thus the system of equations to be solved is

$$\begin{aligned}
\pi_1 q_{11} &+ \pi_2 q_{21} &&+ \ldots + \pi_I q_{I1} &&= 0 \\
\pi_1 q_{12} &+ \pi_2 q_{22} &&+ \ldots + \pi_I q_{I2} &&= 0 \\
&\vdots &&\vdots &&\vdots \\
\pi_1 q_{1,I-1} &+ \pi_2 q_{2,I-1} &&+ \ldots + \pi_I q_{I,I-1} &&= 0 \\
\pi_1 &+ \pi_2 &&+ \ldots + \pi_I &&= 1 ,
\end{aligned}$$

which can be written as

$$B\Pi' = C , \qquad (10)$$

where B is $I\times I$ and is the transpose of Q with the last row replaced by 1's and where Π' is the transpose of Π. Here C is an $I\times 1$ column vector and consists of all zeroes, except for the last element, which is unity. More precisely,

$$B = \begin{pmatrix} q_{11} & q_{21} & \cdots & q_{I1} \\ q_{12} & q_{22} & \cdots & q_{I2} \\ \vdots & \vdots & & \vdots \\ q_{1,I-1} & q_{2,I-1} & \cdots & q_{I,I-1} \\ 1 & 1 & & 1 \end{pmatrix},$$

$$\Pi' = \begin{pmatrix} \pi_1 \\ \pi_2 \\ \vdots \\ \pi_I \end{pmatrix}, \quad C = \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix}.$$

The goal is to obtain the elements π_i. Therefore, directly from (10) we get

$$\Pi' = B^{-1}C .$$

Therefore, by inverting B we can obtain the general time steady state probabilities. An interesting thing occurs because of the form of C. Let the elements of B^{-1} be b_{ij}. Then,

$$\Pi' = \begin{pmatrix} b_{11} & \cdots & b_{1I} \\ b_{21} & & \vdots \\ \vdots & & \\ b_{I1} & & b_{II} \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} b_{1I} \\ b_{21} \\ \vdots \\ b_{II} \end{pmatrix}.$$

In other words, of the whole inverted B matrix, only the last column is needed, and it itself is the solution. This has computational and core storage implications as will be demonstrated below.

One way to invert the B matrix is to first append an identity matrix of the same dimensions to the side, e.g.,

$$\left(\begin{array}{cccc|cccc} q_{11} & q_{21} & \cdots & q_{I1} & 1 & 0 & \cdots & 0 \\ q_{12} & q_{22} & \cdots & q_{I2} & 0 & 1 & & \\ \vdots & \vdots & & \vdots & & & \ddots & \vdots \\ q_{1,I-1} & q_{2,I-1} & \cdots & q_{I,I-1} & & & & \\ 1 & 1 & \cdots & 1 & 0 & & & 1 \end{array}\right),$$

and then to perform a Gauss-Jordan type reduction on the B matrix, while simultaneously performing the same steps on the identity matrix. That is, we start with $B \mid I$ and end with $I \mid B^{-1}$.

However, as was discussed earlier, only the last column of B^{-1} is needed. Thus, the Gauss-Jordan reduction could be carried out on B with only a column vector of dimension $I \times 1$ (consisting of all zeroes except the last element, which is unity) appended to B.

In order to clarify the discussion so far, a simple example will be worked out in detail. Consider a two-stage system with single servers (exponential) at each stage and one customer of each type in the system. The customer of type i has a service rate of μ_{ij} at stage j (i,j = 1,2). The six possible states are:

State					
1	0	0	1	2
2	0	0	2	1
3	0	1	0	2
4	0	2	0	1
5	1	2	0	0
6	2	1	0	0

The possible transitions are:

From State No.	To State No.	Service Rate for Transition
1	4	μ_{22}
2	3	μ_{12}
3	1	μ_{11}
3	6	μ_{22}
4	5	μ_{12}
4	2	μ_{21}
5	3	μ_{21}
6	4	μ_{11}

The intensity matrix, Q, is

$$\begin{pmatrix} -\mu_{22} & 0 & 0 & \mu_{22} & 0 & 0 \\ 0 & -\mu_{12} & \mu_{12} & 0 & 0 & 0 \\ \mu_{11} & 0 & -\mu_{11}-\mu_{12} & 0 & 0 & \mu_{22} \\ 0 & \mu_{21} & 0 & -\mu_{21}-\mu_{12} & \mu_{12} & 0 \\ 0 & 0 & \mu_{21} & 0 & -\mu_{21} & 0 \\ 0 & 0 & 0 & \mu_{11} & 0 & -\mu_{11} \end{pmatrix}$$

The matrix B, i.e., the transpose of Q with the last row all ones, is

$$\begin{pmatrix} -\mu_{22} & 0 & \mu_{11} & 0 & 0 & 0 \\ 0 & -\mu_{12} & 0 & \mu_{21} & 0 & 0 \\ 0 & \mu_{12} & -\mu_{11}-\mu_{22} & 0 & \mu_{21} & 0 \\ \mu_{22} & 0 & 0 & -\mu_{21}-\mu_{12} & 0 & \mu_{11} \\ 0 & 0 & 0 & \mu_{12} & -\mu_{21} & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Assigning the following numerical values for the inversion,

$$\mu_{11} = 2 \ , \quad \mu_{12} = 4 \ , \quad \mu_{21} = 3 \ , \quad \mu_{22} = 4 \ ,$$

the matrix B with the last column of an identity matrix appended is

$$\left(\begin{array}{cccccc|c} -4 & 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & -4 & 0 & 3 & 0 & 0 & 0 \\ 0 & 4 & -6 & 0 & 3 & 0 & 0 \\ 4 & 0 & 0 & -7 & 0 & 2 & 0 \\ 0 & 0 & 0 & 4 & -3 & 0 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{array}\right) .$$

A Gauss-Jordan reduction on this system yields

$$\left(\begin{array}{cccccc|c} 1 & 0 & 0 & 0 & 0 & 0 & .0814 \\ 0 & 1 & 0 & 0 & 0 & 0 & .1047 \\ 0 & 0 & 1 & 0 & 0 & 0 & .1628 \\ 0 & 0 & 0 & 1 & 0 & 0 & .1395 \\ 0 & 0 & 0 & 0 & 1 & 0 & .1860 \\ 0 & 0 & 0 & 0 & 0 & 1 & .3256 \end{array}\right)$$

The last vertical column represents the steady state results. For example, the probability of being in state 3, which is $0_1^1 \mid 0_2^2$ (type 1 customer in service at stage 1 and type 2 customer in service at stage 2), is .1628 . With the general-time steady-state probabilities thus obtained, measures of the system, such as availability, are directly computable. Using the parameters of this example in Equations (2) and (3), we compute an availability of .6190 .

2.6 Computer considerations

There are two principal considerations in implementing this algorithm: execution time in performing the Gauss-Jordan reduction on the B matrix, and core storage for the B matrix.

The execution time can be minimized by taking into account the nondense nature of the B matrix. Since many of the elements are zero, there is no need to operate on such a row when eliminating nonpivot element column elements.

The core storage problem is another story. A system with five customers of each type has an upper bound of $\frac{(5+5+1)!}{5!5!} = 2772$ states.

If that bound were achievable, an array of 2772×2773 real variables would be required (for the B matrix augmented by the last column of an identity matrix). Note that the Q matrix can be defined, transposed, and altered into the B matrix all in the same array to save storage, but this is still unreasonable for in-core storage.

An alternative is to store the augmented B matrix on some mass storage device (e.g., disk or drum) and then only consider two rows at a time in core, the pivot element row and a row to be operated on. This extensive input/output to the mass storage device would hurt execution time, but only requires an in-core array of $2 \times (I+1)$ real variables, where I is the number of feasible states.

Another alternative might be an approach similar to HANNIBALSSON and DISNEY (1977), where by appropriately defining state numbers, the B matrix might be put into a block diagonal form with smaller matrices as elements. This would remove many zeroes arising from the nondense intensity matrix and result in better utilization of computer core. However, these alternatives are not investigated further here.

The computer program written for calculations in this paper had a maximum of 107 feasible states for an in-core array of (107,108). This was sufficient to handle a case with four of type 1, three of type 2, four servers at stage 1, and two servers at stage 2.

3. Approximate Model Development

For each stage the approximation takes a weighted average of service rates of the two classes and uses it for both classes in the standard machine repair problem with spares formulation for one class of customers. More precisely, for N of type 1, M of type 2, μ_{ij} = service rate of type i at stage j, $\bar{\mu}_j$ = approximate service rate for each server at stage j, we have

$$\bar{\mu}_1 = \frac{N\mu_{11} + M\mu_{21}}{N+M}$$

$$\bar{\mu}_2 = \frac{N\mu_{12} + M\mu_{22}}{N+M} \quad . \tag{11}$$

The classic machine repair equations can now be used to determine steady state probabilities. Availabilities are computed in the usual way [see Equation (4)].

4. Numerical Comparisons

4.1 Different rates at stage 1

Figures 7 through 10 show the results of a series of parametric trade-offs, using availability as a measure of the system and comparing the exact and approximate models. Data used in the plots are given in Appendix A. A baseline system is defined with three customers of type 1, three of type 2, four servers at stage 1, and two servers at stage 2. Figure 7 shows a variation of plus or minus one in the number of type 1 customers, all other system parameters being held constant. Figure 8 shows a variation in the number of servers at stage 1; Figure 9 shows a variation in servers at stage 2; and Figure 10 shows a trade-off where the total number of customers is kept constant at six, but the mix between types 1 and 2 is varied.

In all cases in these figures the service rates at stage 2 are identical for either class of customer and the service rates at the first stage are given as a ratio to the second stage rate, i.e.,

$$\rho_1 = \frac{\mu_{11}}{\mu_{12}} \quad \text{and} \quad \rho_2 = \frac{\mu_{21}}{\mu_{22}} \quad \text{and} \quad \mu_{12} = \mu_{22} ,$$

where μ_{ij} is the service rate for type i customer at stage j. The values of ρ_1 for type 1 and ρ_2 for type 2 are varied from .2 to 1.0 in all cases. Consider Figure 7 when $\rho_1 = \rho_2$; the exact and approximate models are equivalent, as they should be. As the difference between ρ_1 and ρ_2 increases, so does the percent error. For the baseline system, the percent error in the approximate model goes up to almost 30% when $\rho_1 = .2$ and $\rho_2 = 1.0$ (or vice versa for $\rho_1 = 1.0$ and $\rho_2 = .2$ because of symmetry; when there are equal numbers of each type of customer, it does not matter which is considered type 1 or 2). Note that the approximate model always underestimates the true availability in Figure 7 (and in all the other figures of this parametric study). This was also observed and proved by

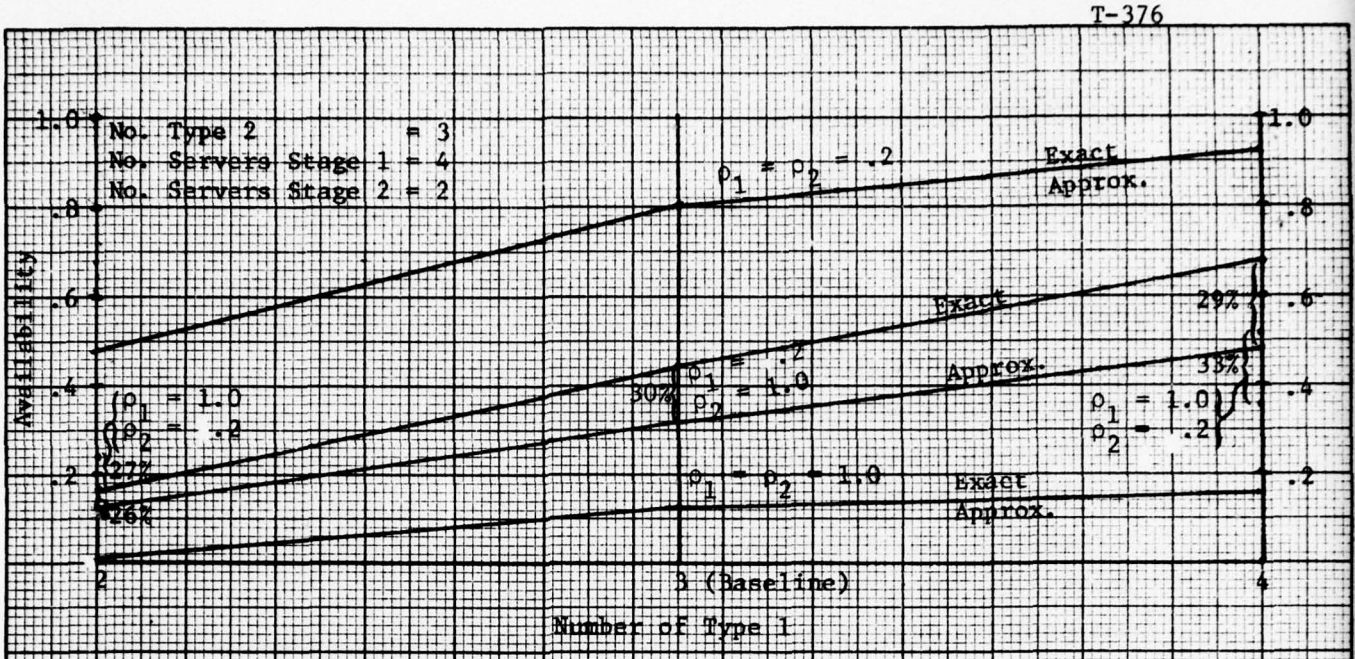


Figure 7. Effect of number of Type 1.

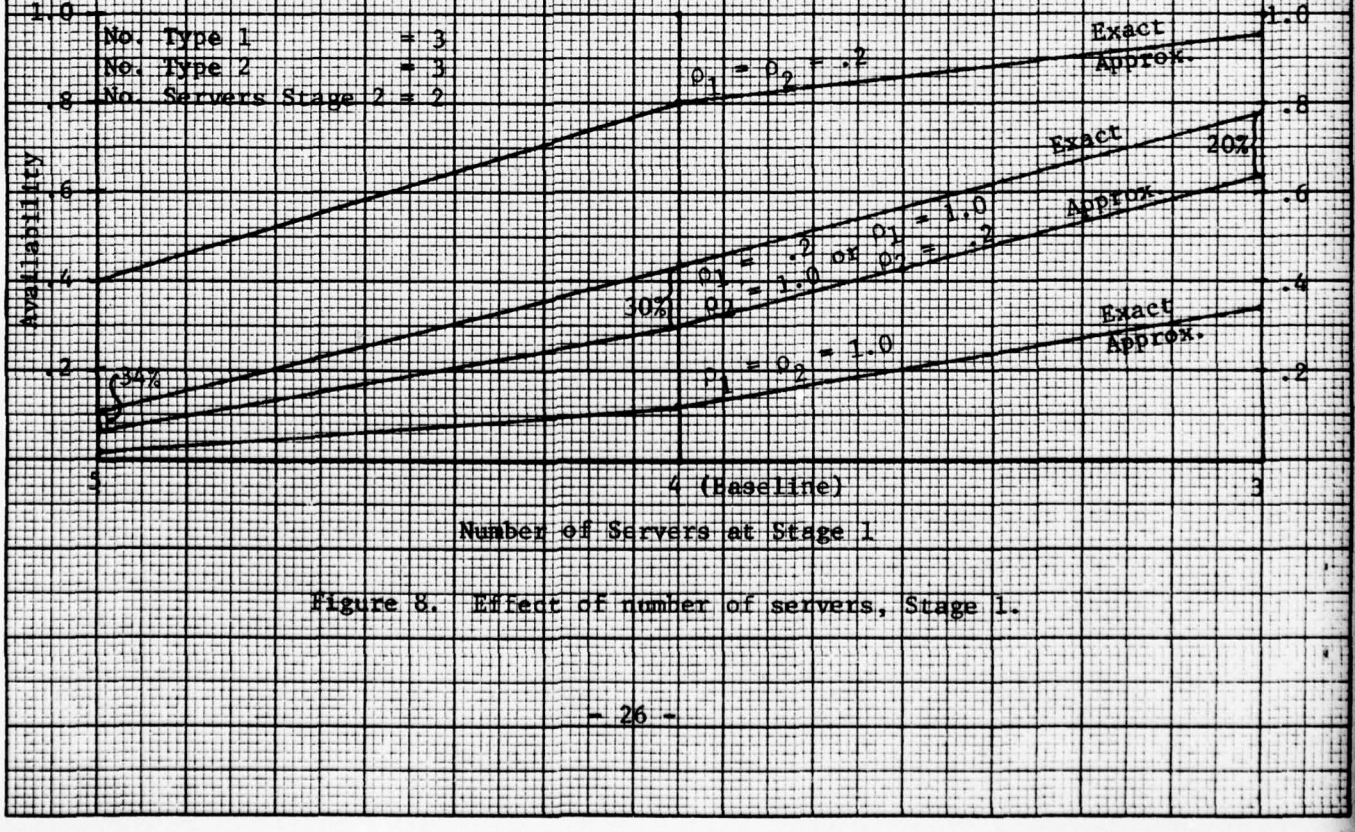


Figure 8. Effect of number of servers, Stage 1.

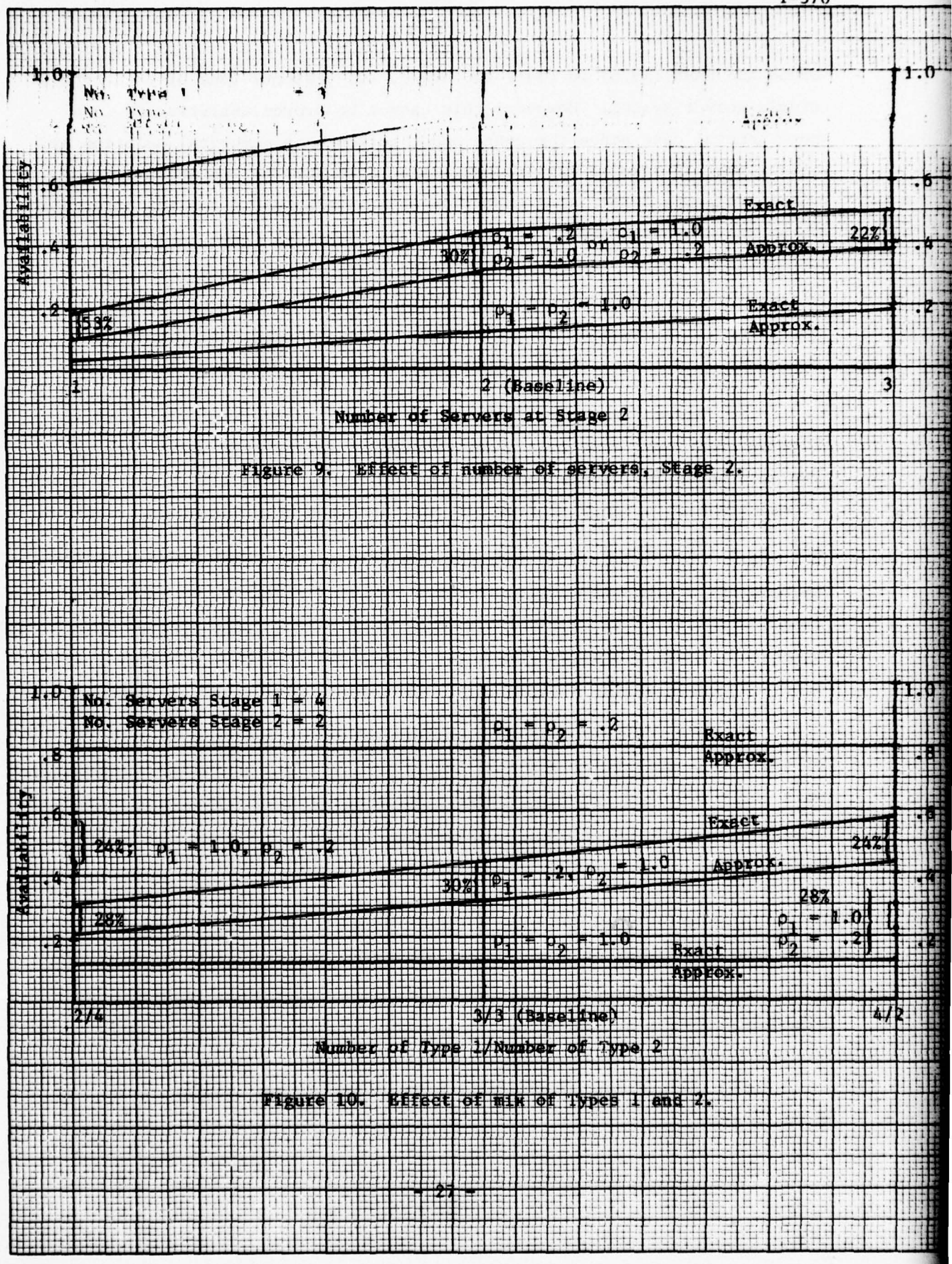


Figure 9. Effect of number of servers, Stage 2.

Figure 10. Effect of mix of Types 1 and 2.

GROSS and KAHN (1976) in their two stage, two customer (one of each type), single server system. However, this cannot be proved analytically in a more general case since the exact model relies on a numerical solution and thus algebraic results are not available to compare to the approximate model's algebra. Note also that this underestimation is for the situation where the approximation is at stage 1, the operating stage, only. The servicing at stage 2 is really not being approximated since both types of customer have the same service rate at stage 2. The approximation of availability becomes an overestimation when stage 2 has unequal rates but stage 1 has equal rates. This will be further discussed below.

In Figure 7 the baseline's availability ranges from .12 to .80. When an additional type 1 is added, resulting in a system whose results are plotted on the right, the availability increases in all cases, as should be expected, to a range of from .15 to .92. When considering $\rho_1 = .2$ and $\rho_2 = 1.0$ versus $\rho_1 = 1.0$ and $\rho_2 = .2$, the former shows a greater increase in exact availability from the baseline (from .44 to .67 versus from .44 to .53). To explain this first note that a lower service rate in stage 1 means a greater time between failures (and thus a better unit) which increases system availability. Therefore, with $\rho_1 = .2$ and $\rho_2 = 1.0$, the more reliable operating unit is of type 1 and this is the type of additional unit being added to the system in the trade-off. Conversely, with $\rho_1 = 1.0$ and $\rho_2 = .2$, type 1 is the poorer unit and an additional poorer unit does not increase availability as markedly as an additional better unit.

Similarly, when removing a type 1 from the system, the availabilities drop to a range of from .05 to .48. With $\rho_1 = .2$ and $\rho_2 = 1.0$, a better unit is being removed and this affects system availability more than when $\rho_1 = 1.0$ and $\rho_2 = .2$ and a poorer unit is being removed.

Similar general observations on the results can be made on Figures 8, 9, and 10.

Figure 8 shows availability increasing with a decreasing number of servers at stage 1 (i.e., a fewer number of machines need to be operating). Since there are equal numbers of types 1 and 2 for all cases, the plots for $\rho_1 = .2$, $\rho_2 = 1.0$ and $\rho_1 = 1.0$, $\rho_2 = .2$ are identical. The baseline case is the same as before and shows a maximum error of almost 30% for the approximate model and 34% and 20% for the trade-offs.

Figure 9 shows availability increasing with an increasing number of servers at stage 2. The approximate model's maximum error goes from 30% in the baseline to 53% and 22% in the two trade-offs.

Figure 10 is slightly different from the others because there is not a true parametric trade-off underway, where only one parameter is varied and all others held constant. In Figure 10 the total number of customers is held constant while the numbers of each type of customer are varied simultaneously. For $\rho_1 = \rho_2$, availability is not affected by variations between the numbers of each type. When $\rho_1 \neq \rho_2$, then the availability increases if one is simultaneously increasing the number of better units (lower ρ) and decreasing the number of poorer units (higher ρ). Conversely, the availability decreases when varying the units the other way.

All of the above observations on the numerical results are consistent with a common sense, qualitative analysis. What the numerical results can add is a quantitative feel for the trends, as well as an assessment of the approximate model. The most important parameter to vary from the baseline in order to change availability is the number of servers at stage 1. This parameter generally yields the steepest slopes. The next most important parameter is adding or subtracting an additional "good" (low ρ) unit. This is followed by varying the number of servers at stage 2. Finally, juggling the numbers of each type while holding their sum constant seems the least effective way to change availability. These are general conclusions based on all of the slopes in Figures 7 through 10. Individual slopes for particular ρ_1, ρ_2 combinations can be found

that do not agree with the comments above. In addition, these conclusions are relative to the baseline as defined and would not necessarily hold for other systems. The parametric study was mainly done in an attempt to demonstrate how the exact and approximate models can be used as tools in analyzing cyclic queueing systems with different classes of customers and also in an attempt to investigate the accuracy of the approximate model which will now be addressed.

The accuracy of the approximate model appears to be more a function of how greatly ρ_1 and ρ_2 differ than by how high is the availability. Gross and Kahn (1976) found that in their two customer, single server system, the approximation was good under a high availability constraint. That is not necessarily inconsistent with these results, just not as general.

Considering that Gross and Kahn proved for their system that the approximate model was always a lower bound on the true availability, if one were to impose a "high" availability constraint on the approximate model, the region in which the exact availability could fall would be compressed (i.e., between the "high" approximate results and 1.0) and the percent error limited. For example, if one varied system parameters until the approximate model yielded a .95 availability, the maximum possible error would be 5%. An examination of the Gross and Kahn data also supports the conclusion that if the maximum absolute difference between ρ_1 and ρ_2 is limited to .6, the error is again less than 5%, but independent of the availability (which goes as low as .45).

The highest availability obtained in the trade-off in this paper was .96 in Figure 8 for $\rho_1 = \rho_2 = .2$, at which condition there is 0% error since all the units are identical. Now consider the following trade-off: holding ρ_2 constant at .2, decrease ρ_1 in steps from .2 to .001; repeat the ρ_1 variations for ρ_2 = constant at .4 and then 1.0. The results are shown in Table I. For a high availability constraint on the *approximate model*, note the results for $\rho_2 = .2$ and .4 where the

TABLE I

ACCURACY OF APPROXIMATE MODEL AS FUNCTION OF AVAILABILITY

ρ_1	ρ_2	$AVAIL_{Exact}$	$AVAIL_{Approximate}$	% Difference
.2	.2	.9643	.9643	0
.1	.2	.9853	.9832	.2
.05	.2	.9937	.9898	.4
.01	.2	.9987	.9937	.5
.001	.2	.9996	.9944	.5
.2	.4	.9157	.9039	1.3
.1	.4	.9618	.9376	2.5
.05	.4	.9817	.9519	3.0
.01	.4	.9948	.9620	3.3
.001	.4	.9972	.9641	3.3
.2	1.0	.7828	.6301	19.5
.1	1.0	.8838	.6774	23.4
.05	1.0	.9334	.7014	24.9
.01	1.0	.9699	.7207	25.7
.001	1.0	.9776	.7521	25.8

Number of Customers of Type 1 = 3

Number of Customers of Type 2 = 3

Number of Servers at Stage 1 = 3

Number of Servers at Stage 2 = 2

$$\rho_1 = \frac{\mu_{11}}{\mu_{12}}$$

$$\rho_2 = \frac{\mu_{21}}{\mu_{22}}$$

$$\mu_{12} = \mu_{22}$$

μ_{ij} = service rate of type i customer at stage j; i,j = 1,2 .

error is less than 4%. As discussed earlier, the high approximate availability apparently bounds the exact from below and thus limits the possible error. However, if the high availability constraint were imposed on the true system (i.e., the *exact model*), errors up to 26% could be realized. Naturally, if one can compute the exact availability, upon which to place the constraint, it is not necessary to discuss errors from an approximate model. Thus, the point here is that when one speaks of imposing a high availability constraint in order to justify the use of the approximate model in place of the exact (which may not be obtainable because of considerations like too vast a state space), then it must be understood that the constraint is on the approximate model's results.

Table I again supports the observation that the accuracy of the approximate model is really a function of the difference between ρ_1 and ρ_2, independent of availability (except by virtue of being bounded from below at high approximate model availabilities). We will now attempt to put some limits on the error as a function of the ρ_1, ρ_2 difference.

In Appendix A, for an absolute difference of .2 (e.g., for $\rho_1 = .6$ and $\rho_2 = .8$), the maximum error for the approximate model is 8%, with most under 3%. If we consider an absolute difference of .4, the maximum error is 24%, with most under 10%. However, these maxima occur at conditions where $\rho_1 = .2$, $\rho_2 = .4$ and $\rho_1 = .2$, $\rho_2 = .6$, so in a multiplicative sense (versus an absolute difference), these conditions are more extreme than, say, $\rho_1 = .6$, $\rho_2 = .8$ and $\rho_1 = .6$, $\rho_2 = 1.0$.

Therefore, at least for the parameters of this system, an absolute difference between ρ_1 and ρ_2 of .2 would be a reasonable range to assign for accuracy of the approximate model. Then, independent of the true system availability, the approximate model would underestimate the exact system availability by no more than 8%. In most realistic applications,

differences in units will not be great as gradual technological learning usually takes place.

4.2 Different rates at stage 2

The above numerical results were all obtained holding the service rates at stage 2 constant and equal for both types of customers. An investigation is now made for the opposite situation: assuming both classes of customers have the same operating time distribution at stage 1, but have different servicing rates at stage 2.

Table II shows the results of this limited investigation. The parameter varied is the number of servers at stage 2. There are three customers of each type in the system. Both classes have a service rate of 1.0 at stage 1. Customers of type 1 have a service rate of 5.0 at stage 2 and customers of type 2 a rate of 1.0. Thus, $\rho_1 = .2$ and $\rho_2 = 1.0$. The exact system availabilities range from .14 to .45, but the important observation is that now the approximate model overestimates the true availability (from 45% to 123%).

4.3 Impact of averaging service rates on system flow rate

Before trying to explain the under- or overestimation resulting from averaging service rates at the first or second stage, consider the following. Let there be an infinite queue in front of a single-server, one-stage system, where the queue consists of alternating type 1 and type 2 customers. The average output rate from the single server would then be the inverse of the average service times for the two types, where each service time itself is the inverse of that type's service rate. For example, if $\mu_1 = 2$ service completions per second and $\mu_2 = 3$ service completions per second (where μ_i = service rate of type i, i=1,2), then the average output rate from the single server would be $1/\{(1/2 + 1/3)/2\} = 2.4$ customers per second, i.e., an equivalent one stage system in terms of

TABLE II

ACCURACY OF APPROXIMATE MODEL--DIFFERENT SERVICE RATES AT STAGE 2

No. Servers at Stage 2	Exact Availability	Γ Exact	Approximate Availability	Γ Approximate	% Difference
1	.1402	1.6825	.3124	2.7262	-122.8
2	.4001	2.9884	.6077	3.6339	- 51.9
3	.4497	3.5142	.6496	3.7992	- 44.5

For All Cases:

Number of Servers at Stage 1 = 4

Number of Customers of Type 1 = 3

Number of Customers of Type 2 = 3

$\mu_{11} = 1.$

$\mu_{21} = 1.$

$\mu_{12} = 5.$

$\mu_{22} = 1.$

$\rho_1 = .2$

$\rho_2 = 1.0$,

where μ_{ij} = service rate of type i at stage j; i,j = 1,2 ,

and $\rho_1 = \frac{\mu_{11}}{\mu_{12}}$

$\rho_2 = \frac{\mu_{21}}{\mu_{22}}$

Γ = expected cycle flow rate.

output rate would be a system with only one class of customers with a service rate of 2.4.

Now consider a system with a large number of servers, each always busy, outputting only one class of customer, either type 1 or type 2. The overall output rate from that multiserver stage is equivalent to the overall output rate of a system where each server has the average service rate of the original system. For example, assume N servers are outputting type 1's and N servers outputting type 2 customers. The overall output rate from that stage is $N\mu_1 + N\mu_2$. An equivalent system would have each server with a rate of $(\mu_1 + \mu_2)/2$ and the 2N servers would thus have the same overall output rate, $N(\mu_1 + \mu_2)$, as the original system. For the above numerical example ($\mu_1 = 2$ and $\mu_2 = 3$) the average output rate per channel is now 2.5.

The point of the above discussion can be summarized by saying that averaging service times implies a single server, and averaging service rates implies infinite servers.

Now we will return to the discussion of the underestimation of availability (due to averaging service rates at the first stage) and overestimation of availability (due to averaging service rates at the second stage) by considering the expected flow through the system, Γ. Since the system is only considered in steady state, Γ is both the expected flow of failures out of stage 1 and the expected flow of repaired units out of stage 2, and is given by

$$\Gamma = \sum_i \Pr\{\text{system in state } i\}(n_i\mu_{11} + m_i\mu_{21}) ,$$

where n_i, m_i represent the number of type 1 and 2 customers, respectively, *in service* (as opposed to in the queue) at stage 1 for state i, and μ_{ij} is the service rate of type i customer at stage j.

When one averages the service rates at stage 1, Γ of the system is increased. The approximate model that was described above averages

rates, and this implies infinite servers, which increases customer throughput at that stage, i.e., the system flow is "speeded up" at the expense of congestion at stage 1. Hence, availability is lowered.

For different service rates at stage 2, a similar argument can be made for the approximate model reducing congestion at the *second stage* by speeding up the system flow. However, this allows relatively more congestion at the first stage and hence a higher availability than the exact model. This higher flow rate and higher availability can be seen numerically in Table II, where the flow rates are denoted by Γ exact and Γ approximate. In other words, averaging at either stage speeds up the flow through the system. If the speed-up is at the expense of first stage congestion (i.e., first stage service rate averaging), the availability will drop; if it is at the expense of second stage congestion (i.e., second stage service rate averaging), the availability will increase.

5. An Approximate Model Based on Averaging Service Times

The preceding discussion referenced an analogy wherein averaging service times implies a single server. This suggests another approximate model using

$$\bar{\mu}_1 = \left[\frac{\frac{N}{\mu_{11}} + \frac{M}{\mu_{21}}}{N+M}\right]^{-1}, \qquad \bar{\mu}_2 = \left[\frac{\frac{N}{\mu_{12}} + \frac{M}{\mu_{22}}}{N+M}\right]^{-1}. \tag{12}$$

If averaging rates and times implies infinite servers and a single server, respectively, an appealing idea is to see if the two approximations bound an actual case with an arbitrary number of servers in an exact analysis. Appendix B contains tabulated data for runs made to test this hypothesis. The tabulation includes results for flow rates, spares availability (i.e., the conditional probability that a spare is available given

that a failure is about to take place), and a general time (virtual) availability (i.e., the probability that a spare is in inventory at any arbitrary time) for three models: the approximation based on averaging rates, the exact analysis, and the approximation based on averaging times.

Based on the cases presented in Appendix B, and on all other runs made during the course of this research, the following assertions can be made (with one exception, to be discussed later):

1. The availability and general time availability are bounded by the two approximations.
2. The exact flow rate is bounded by the approximations.
3. For different service rates at stage 1 and the same at stage 2, relative to the exact model,
   a. averaging rates speeds up the system flow at the expense of congestion at stage 1 and thereby decreases availability and general time availability,
   b. averaging times slows up the system flow by increasing congestion at stage 1 and thereby increases availability and general time availability.
4. For different service rates at stage 2 and the same at stage 1, relative to the exact model,
   a. averaging rates speeds up the system flow at the expense of congestion at stage 2 and thereby increases availability and general time availability,
   b. averaging times slows down the system flow by increasing congestion at stage 2 and thereby decreases availability and general time availability.

The only numerical exception encountered to the above statements occurred when there was only one customer of each type in the system. In that case the exact availability was still bounded (as analytically proved in Appendix C), but the exact flow rate and general time availability were not (just slightly outside of the averaging time approximation).

Further research is warranted in this area of analyzing the two approximations as bounds. Analytically, it is untenable, as formulated herein, because of the matrix relationships that do not lend themselves to algebraic manipulations. However, using the concept of stochastically monotone Markov chains [see DALEY (1968)], provable bounds for the exact model using the two approximate models might be achievable under certain conditions. However, for the purposes of this paper, the approximate models can only be considered to yield "approximate" bounds for systems characterized by the range of parameters numerically investigated here.

6. Summary of Numerical Results

For different customer service rates at stage 1 and the same at stage 2, the following holds:

$$\text{AVAIL}_{\text{Average rates}} \leq \text{AVAIL}_{\text{Exact}} \leq \text{AVAIL}_{\text{Average Times}}, \quad (13)$$

where the subscript "average rates" means a relationship as in Equation (11) and "average times" means a relationship as in Equation (12).

For different customer service rates at stage 2 and the same at stage 1, (13) holds with the inequalities reversed.

Extending the above assertions, one can reason the following for a system with two classes of customers where each has a different rate at each stage. Let one approximation be made by using average rates at stage 1 and average times at stage 2. This should yield a lower bound on the exact availability. Let the other approximation be made by using average times at stage 1 and average rates at stage 2. This should yield an upper bound.

Furthermore, one could reason the following for a general multi-stage, multiclass cyclic queue where each customer class has its own service rate at each stage. Let one approximation be made by using average rates at the first stage (thus speeding up the flow through stage 1 and reducing availability) and using average times at all other stages (thus slowing down the flow through all other stages and reducing availability).

Let the other approximation be made by using average times at the first stage and average rates at all others (thus increasing availability). Then, these two approximations should bound the exact availability, and the cyclic queueing theory results for homogeneous customers [see Gross and Ince (1978)] can be used for obtaining the approximate answers.

7. Priorities

Section 2 of this paper dealt with a first come-first served (FCFS) discipline, i.e., customers queued up in order of arrival to a stage and entered service in that order. A priority service discipline can easily be established for the exact model wherein if there is a queue with both customer classes present at either stage, one particular class (the "privileged" class) will always be chosen over the other as servers become available. Only if the privileged class is not in the queue will the other class be accepted into service.

Note that customers of the nonprivileged class are allowed to complete service once they have started, even if a privileged class customer subsequently joins the queue. There are service disciplines that allow service interruptions, but such a preemptive priority discipline will not be considered here.

It was stated above that a priority discipline could be easily incorporated. This is because once all the logic of determining feasible states, transitions, the intensity matrix, inversions, etc., has been established for the FCFS model, it is only a matter of filtering the feasible unique states such that orderings that show nonpriority customers ahead of priority customers in a queue are considered nonfeasible and thus are dropped. For example, assuming that type 2 has priority over type 1 in both stages, consider a system with three customers of type 1, four of type 2, four servers at stage 1, and two servers at stage 2. A sample of orderings generated by the algorithm is shown in Table III.

In total, the nonpriority system would have 107 feasible unique states. The priority system would have only 47 states. With the list of

TABLE III

SAMPLE CUSTOMER ORDERINGS SHOWING EFFECT OF PRIORITY

System State				Without Priority	With Priority
000	0002	02221	11	feasible	unfeasible
000	0011	00122	22	feasible	feasible
000	0011	00212	22	feasible	unfeasible
000	0011	00221	22	feasible	unfeasible
000	0011	00222	12	feasible	feasible
000	0012	00112	22	feasible	feasible
000	0012	00121	22	feasible	unfeasible

feasible states generated for the priority system, transitions are determined in the usual way, except that after an ordering has been advanced to reflect a service completion, customers in the queue are adjusted to the lowest base 3 number to reflect the type 2 priority over type 1 in the queue. Note that for the algorithm of this paper, the priority class of customer is always type 2.

It would seem that a system with priority would generally have either a much higher or lower (depending on which are the better units) availability than one without. However, the difference is very dependent on the system parameters. In the above sample system, where μ_{ij} is the service rate for type i customer at stage j, and for $\mu_{11} = 1.0$, $\mu_{12} = 1.0$, $\mu_{21} = .2$, $\mu_{22} = 1.0$, the exact availability without priority is .67 and with priority is .70 (for priority of type 2 over type 1).

7.1 Numerical comparisons

Figures 11 through 13 and Table IV show three series of trade-offs on this system, varying first the number of type 2 customers, then the number of servers at stage 1, and finally the number of servers at stage 2. Data for these figures are contained in Table IV. The changing from FCFS

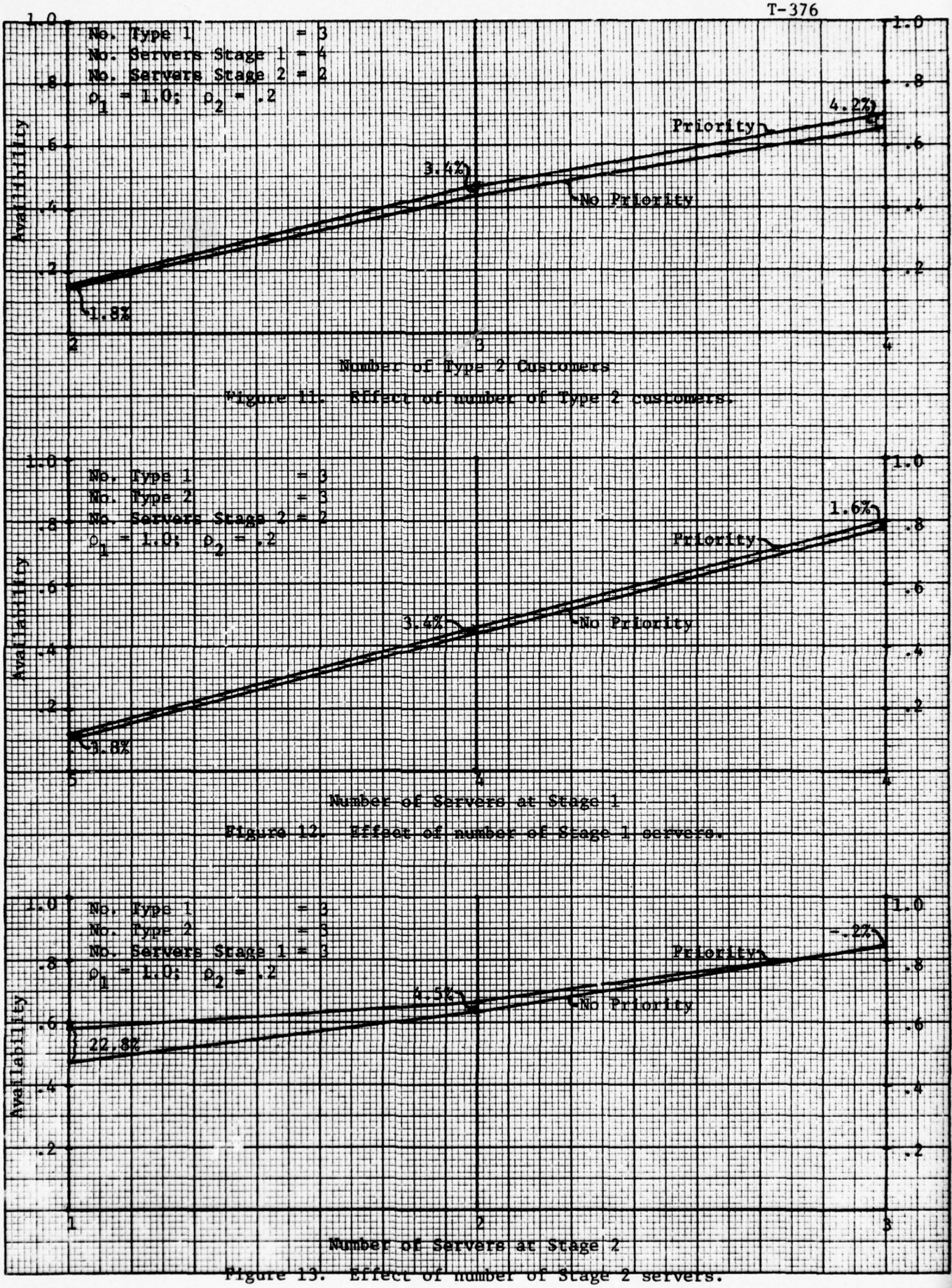


Figure 11. Effect of number of Type 2 customers.

Figure 12. Effect of number of Stage 1 servers.

Figure 13. Effect of number of Stage 2 servers.

TABLE IV

DATA FOR PRIORITY RUNS

Number of Type 1 Customers	Number of Type 2 Customers	Number of Servers at Stage 1	Number of Servers at Stage 2	Availability Without Priority	Availability With Priority	Percent Difference
3	4	4	2	.6711	.6992	4.2
3	3	4	2	.4444	.4596	3.4
3	2	4	2	.1499	.1526	1.8
3	3	2	2	.9661	.9678	.2
3	3	3	2	.7828	.7954	1.6
3	3	4	2	.4444	.4596	3.4
3	3	5	2	.1141	.1184	3.8
3	3	3	3	.8530	.8511	- .2
3	3	3	2	.6426	.6716	4.5
3	3	3	1	.4749	.5832	22.8

to priority of type 2 is shown to increase availability less than 5% in the first two series and almost up to 23% in the third for a single server at stage 2. However, in the third series of trade-offs (varying the number of servers at stage 2), availability is seen to actually *decrease* because of priority when the number of servers at stage 2 is increased to three. This counter-intuitive result will be discussed further below. In Figure 11, increasing the number of type 2 customers from two to three to four increases the availability of a nonpriority system because an additional customer is being added (and also the additional customer is the better of the two types because it has the lower ρ). Adding priority of type 2 over type 1 to this system then results in an across-the-board increase in availability as shown, albeit only slightly (from 1.8% to 4.2%).

In Figure 12, for a system without priority, decreasing the required number of machines to be operating results in an increase in availability. Superimposing priority on this system slightly increases availability. However, the percent increase is decreasing from 3.8% to 1.6%. Although not plotted, the next step with only two servers at stage 1 would result in an availability of .966 for the nonpriority system versus .968 for the priority system, only a .2% increase.

Figure 13 actually shows a crossover where priority penalizes the system. For a system without priority, increasing the number of servers at stage 2 increases availability as customers are serviced more promptly and returned to the operating stage. Adding priority at first increases availability by almost 23% when there is only one server at stage 2, but the difference decreases rapidly until priority actually *reduces* availability when there are three servers at stage 2.

This apparent anomaly can be explained by considering the fact that availability, as defined here, is a *conditional probability*--the probability that a spare is available given that a failure is about to take place. This is different from a general time probability that a spare is available (i.e., the queue at stage 1 is nonzero at any arbitrary point in time). Logic would lead one to conclude that whenever

the class of better units (lower ρ) is allowed both to operate in stage 1 and be serviced in stage 2 ahead of the poorer units, the percent of time (i.e., the general time probability) there is congestion at stage 1 would be greater than without priority. This was in fact observed in all priority cases run. In Figure 13, for three servers at stage 2 the general time availability for the priority system is .8608 versus .8419 for the nonpriority system. However, when the availability measure is conditionalized, the priority system drops to .8511 while the nonpriority system rises to .8530, thus causing the observed crossover.

Such a phenomenon might be explained "mathematically" by the following: in comparison to a nonpriority system the most probable states in a priority system are those with a proportionally larger number of type 2 units in service at stage 1. However, the overall failure rate for these states would be proportionally lower, since the type 2 units have lower ρ. Therefore, the numerator in the availability expression, Equation (3), might be affected more than the denominator to result in a lower conditional availability for a priority system versus a nonpriority system. A physical interpretation of this phenomenon is discussed below.

7.2 Spares availability as an adequate measure of systems performance

The fact that in some cases spares availability goes down with preference given to "better" units calls into question the adequacy of using such a measure for system service. For a physical explanation of why this happens, consider looking at the expected flow through the system, Γ, which was observed in all cases to decrease with priority on the "better" (lower failure rate in the first stage) class of unit, whether the availability went up or down. The significance of this may be illustrated by the following extreme (hypothetical) example: consider a system which had 20 failures in one week and in all but one case had a spare on hand to replace the failing units. Now consider a change to that system, say the introduction of priority for the longer operating units, that results in ten failures in the next week and in all but one case a

spare was on hand. The first system had an availability of .95 and the second .90. Hence, priority has lowered the availability of the system. However, is the system better or worse with the failure rate (Γ) reduced from 20 to ten per week? The answer might depend on who was evaluating the system. For example, the storeroom (spares pool) manager would probably use availability, for he could say that 95% of his customers are satisfied. On the other hand, the overall system manager is interested in how many shortages per unit time occur or how many machines are operating at any time, etc., and would of course prefer the results obtained by introducing priorities, even if availability might drop.

The point to be kept in mind is that availability per se might not be a good system measure, even though it is commonly used. In the extreme example above, where the availability dropped from .95 to .90 with the introduction of priority, the fact that the expected system failure rate was halved might result in that measure being used instead. However, note that since we have a cyclic system, the expected system failure rate is also equal to the expected system repair rate. By introducing fewer and slower repair channels we can also reduce the failure rate--by seldom having any machines operating! Clearly, then, the measure of effectiveness or constraint to be imposed on the system must be chosen carefully for a given application.

Another measure of system performance that might be considered is the general time probability that a spare is present (a virtual availability as opposed to the conditionalized actual availability). A meaningful constraint that might be used in place of "spares availability" is to require that a certain percentage of machines be "up" with some given probability, e.g., $\Pr(\alpha\%$ of the machines are operating$) \geq \beta$. This constraint can be considered as a measure of *machine availability*.

Table V isolates the impact of priority on each stage (the computer program used for this study currently has a logic which assumes that priority on type 2 units will be enforced at both stages). In the first example, ample servers (six) are provided at stage 2, so priority is only effective at stage 1. In the priority case for this example, the effective failure rate, Γ , is very close to μ_{21} , indicating that type 1

TABLE V

IMPACT OF PRIORITY ON EACH STAGE

[μ_{ij} = service rate of type i at stage j]

	Example No. 1		Example No. 2	
Priority?	No	Yes	No	Yes
No. Servers at Stage 1	1	1	5	5
No. Servers at Stage 2	6	6	1	1
No. Type 1	3	3	3	3
No. Type 2	3	3	3	3
μ_{11}	1.0	1.0	1.0	1.0
μ_{21}	.2	.2	.2	.2
$\mu_{12} = \mu_{22}$	1.0	1.0	1.0	1.0
Γ_{Exact}	.3389	.2032	.9836	.9770
Actual Spares Availability	.99989	.99982	.04532	.06257
General Time Spares Availability	.99991	.99995	.01643	.02301

customers seldom enter service at stage 1 because of type 2 priority.

In the second example, there are ample servers (five) at stage 1 (if ample servers had been used, the availability would have been zero by definition) and the congestion is mainly at stage 2. Hence, the effective service rate approaches the service rate at stage 2, where $\mu_{12} = \mu_{22} = 1.0$, for both nonpriority and priority cases.

In both examples of Table V, priority decreases the expected flow rate of the system and increases the general time probability that a spare is available. However, spares availability decreases in one example

(however slightly) and increases in the other when priority is introduced. Although Table V only contains two examples, all numerical results observed were consistent with the above comments.

7.3 Summary

Introducing a priority discipline for type 2 over type 1 did not yield results readily predictable, although in retrospect entirely logical. Generally, for the system parameters used in the trade-offs in the numerical comparisons of this section, priority resulted in only marginal improvements in spares availability. One case did result in a significant increase, but another resulted in a decrease. This brought up the issue of the suitability of using spares availability as a service performance level criterion and suggested that other measures based on general time probabilities might be preferable. The models developed here can easily handle this type of criterion.

8. Extensions

In this section we indicate how the basic two stage, two customer class model can be extended.

8.1 Several stages with two classes

The two stage model can conceptually be easily extended to many stages. The main problems would be practical instead of theoretical (e.g., computer core storage and execution time).

Consider a three stage system with N of type 1 and M of type 2. There must be $N+M$ slots in front of each of the three stages and a state description would consist of $3(N+M)$ integers, versus the $2(N+M)$ of the two stage model. There would be many more possible states and upper and lower bounds could be determined in a similar way as for the two stage case. For a three stage system,

$$\text{Upper Bound (single server at each stage)} = \frac{(N+M+2)!}{N!M!2!}$$

$$\text{Lower Bound (ample servers at each stage)} = \binom{N+2}{N}\binom{M+2}{M},$$

and the expression for the actual number would be similar to Equation (8) with a double summation over three factors. Using the same numerical example as in the two stage system (N=5, M=2) , the lower bound is 126 and the upper is 756 (versus 18 and 168 for the two stage system). These bounds are obviously not very tight. The generation of states would be similar to the two stage system, a base 3 count up, ignoring zeroes, but now through each of the three stages. Transitions would be more bulky to record since a given state could conceivably transition to up to six other states by having at least one of each type of customer in service at all three stages.

From there the treatment of the states, creation of an intensity matrix, inversion, calculation of availability, etc., would be almost identical to the two stage system. The problem would lie mainly in allocating core for the intensity matrix, which may be treatable by some of the alternatives discussed in Section 2.6 concerning mass storage or diagonal matrix formulations.

We can now generalize, in theory, to K stages with two classes of customers. For purposes of comparison, the two-stage results will be noted also.

(a) First feasible state with, say, three customers of each type and two servers at each stage;

2 stages: 0000 ¦ 00 | 1112 ¦ 22

K stages: $\underbrace{0000 \text{ ¦ } 00 \text{ | } 0000 \text{ ¦ } 00 \text{ | } \ldots \text{ |}}_{\text{K-1 Stages}}$ $\underbrace{1112 \text{ ¦ } 22}_{\text{Kth Stage}}$

(b) Upper limit on the number of feasible unique states assuming N of type 1 customer and M of type 2 (based on single servers at each stage);

2 stages: $\frac{(N+M+1)!}{N!M!1!}$

K stages: $\frac{(N+M+K-1)!}{N!M!(K-1)!}$

(c) Lower limit on the number of feasible unique states (based on ample servers at each stage);

2 stages: $(N+1)(M+1)$

K stages: $\binom{N+K-1}{N}\binom{M+K-1}{M}$

(d) Maximum possible transitions from one state;

2 stage: 4

K stage: 2K

(e) Intensity matrix, transposition and alteration and inversion of it, calculation of availability;

2 stage: as described in Section 2

K stage: similar to 2 stage treatment

8.2 Several classes with two stages

An extension from two classes of customers to r classes is straightforward. Again, the primary considerations are computer core storage and execution time. In front of each of the two stages there will be $\sum_{i=1}^{r} N(i)$ slots, where $N(i)$ is the total number of customers of type i in the system. A slot can contain 0 (for empty) or any integer from 1 to r (representing the class of customer in the slot).

The upper bound on the number of feasible unique states is

$$\frac{\left[1 + \sum_{i=1}^{r} N(i)\right]!}{N(1)!\ N(2)!\ \ldots\ N(r)!} .$$

The lower bound is

$$\{N(1)+1\}\ \{N(2)+1\}\ \ldots\ \{N(r)+1\} = \prod_{i=1}^{r} [N(i)+1] .$$

The generation of states would be by base $r+1$ counting, ignoring zeroes, through each of the two stages.

A given state could possibly transition to up to $2r$ other states if there is at least one customer of each class in service at both stages. From there, the treatment of the states, creation of an intensity matrix, etc., are similar to the two class analysis.

8.3 Several classes with several stages

Combining the results of the last two sections now results in the following generalized model:

r = number of classes of customers

$N(i)$ = number of customers in the system of class i; $i=1,\ldots,r$

K = number of stages

c_j = number of parallel servers at stage j; $j=1,\ldots,K$

μ_{ij} = service rate of customer of class i at stage j; $i=1,\ldots,r$; $j=1,\ldots,K$

(a) Number of slots in front of each of the K stages = $\sum_{i=1}^{r} N(i)$

(b) Contents of a slot = $0,1,\ldots,r$ (representing the class of the customer occupying the slot; 0 means empty)

(c) State description $K \sum_{i=1}^{r} N(i)$ integer string of 0's,...,r's

(d) Upper bound on the number of feasible unique states (assuming single servers at all stages):

$$\frac{\left[K - 1 + \sum_{i=1}^{r} N(i)\right]!}{N(1)!\ N(2)!\ \ldots\ N(r)!\ (K-1)!}$$

(e) Lower bound on the number of feasible unique states (assuming ample servers at all stages):

$$\binom{N(1) + K-1}{N(1)}\binom{N(2) + K-1}{N(2)} \cdots \binom{N(r) + K-1}{N(r)}$$

(f) Maximum possible transitions from one state = Kr

(g) All else (intensity matrix, inversion, etc.) similar to the development for the two stage, two classes of customer model.

8.4 Preemptive discipline in queue

In this discipline a privileged class, say type 2, upon entering a stage and finding no servers available, but finding a type 1 in service, preempts the type 1 out of service and into the queue. Thus, not only will type 2 customers always be at the head of the queue, but they will also always be in service unless the stage contains only type 1 customers. Since the service distribution is exponential, once a type 1 customer is removed from service he gets no credit for time already spent in service, and when he subsequently reenters service, he must start completely over.

As in the priority, nonpreemptive model, incorporation of such a discipline would not be difficult once the overall logic of a nonpriority system with different classes has been programmed. In the two class system with type 2 being privileged, for a preemptive discipline feasible states would be restricted to the lowest base 3 number in each stage.

For example, considering the same parameters used in Table III (three customers of type 1, four of type 2, four servers at stage 1, and two servers at stage 2), a similar sample of orderings generated by the algorithm is shown in Table VI.

With the list of feasible states so defined, and transitions limited to states on that list, the treatment from there would be similar to the basic heterogeneous model previously described.

8.5 State dependence of service times; blocking

The analysis so far has considered parallel service channels at the stage, and this is a form of state dependence. The parallel service channels can be represented by

TABLE VI

SAMPLE ORDERINGS SHOWING EFFECT OF PREEMPTIVE PRIORITY

System State				Without Priority	With Non-preemptive Priority	With Preemptive Priority
000	0002	02221	11	feasible	unfeasible	unfeasible
000	0011	00122	22	feasible	feasible	feasible
000	0011	00212	22	feasible	unfeasible	unfeasible
000	0011	00221	22	feasible	unfeasible	unfeasible
000	0011	00222	12	feasible	feasible	unfeasible
000	0012	00112	22	feasible	feasible	feasible
000	0012	00121	22	feasible	unfeasible	unfeasible

$$\text{overall service rate of type } i \text{ at stage } j \text{ for state } k = n_{ij}(k)\mu_{ij} ,$$

where

$n_{ij}(k)$ = the number of type i customers in the service channels at stage j for state k ,

and

$i = 1,2$

$j = 1,2$

$k = 1,2,\ldots,$maximum number of feasible states.

In determining transitions in the basic heterogeneous model, both the state number to which one is transitioning (the match number) and the number of customers in service of the type associated with the particular service completion were recorded. This information was used in setting up the intensity matrix. It would be a straightforward matter to insert one more step between recording the match number and the number of customers in service of a certain type, and setting up the intensity matrix. This step would determine a service rate as some arbitrary function of the number of customers in service of a certain type. This service rate would then be used in the intensity matrix.

In fact, this service rate can be a function of the entire state, not just of the customers in service. In such a way, blocking can easily be incorporated (with an accompanying filtering of feasible states to disallow states made unfeasible because of blocking restrictions) as well as models where servicing rates are stepped up when queues start forming. For example, in our machine repair problem with spares and two different classes of customers, servicing at stage 2 could be increased when failed machines start queueing up and the spares inventory is empty, as well as machine repair ceasing when inventory levels rise over some level.

REFERENCES

BARLOW, R. E. (1962). Repairman problems. *Studies in Applied Probability and Management Science*, 18-33. Stanford University Press.

BASKETT, F., K. M. CHANDY, R. R. MUNTZ, and F. G. PALACIOS (1975). Open, closed, and mixed networks of queues with different classes of customers. *J. Assoc. Comput. Mach.* 22 (2) 248-260.

DALEY, D. J. (1968). Stochastically monotone Markov chains. *Z. Wahrscheinlichkeitstheorie und Verw. Gebiete* 10 305-317.

GROSS, D. and C. M. HARRIS (1974). *Fundamentals of Queueing Theory.* Wiley, New York.

GROSS, D. and J. F. INCE (1978). Spares provisioning for repairable items: cyclic queues in light traffic. To appear in *AIIE Trans.*

GROSS, D. and H. D. KAHN (1976). On the machine repair problem with spares under unequal failure rates. Technical Memorandum Serial TM-66451, Program in Logistics, The George Washington University.

HANNIBALSSON, I. and R. L. DISNEY (1977). An M/M/1 queue with delayed feedback. *Naval Res. Logist. Quart.* 24 (2) 281-291.

INCE, J. F. (1978). Spares provisioning for repairable items. D.Sc. dissertation, The George Washington University.

KELLY, F. P. (1975). Networks of queues with customers of different types. *J. Appl. Prob.* 12 542-554.

POSNER, M. and B. BERNHOLTZ (1967). Two stage closed queueing systems with time lags. *CORS J.* 5 82-99.

POSNER, M. and B. BERNHOLTZ (1968a). Closed finite queueing networks with time lags. *Operations Res.* 16 962-976.

POSNER, M. and B. BERNHOLTZ (1968b). Closed finite queueing networks with time lags and with several classes of units. *Operations Res.* 16 977-985.

APPENDIX A

DATA TABLES FOR FIGURES 7 THROUGH 10

ρ_1	ρ_2	.2	.4	.6	.8	1.0
.2	Exact Avail.	.80461	.67571	.57795	.50299	.44436
	Approx. Avail.	.80461	.65559	.51828	.40336	.31211
	% Difference	0.	3.0	10.3	19.8	29.8
.4	Exact Avail.		.51828	.41308	.33976	.28663
	Approx. Avail.		.51828	.40336	.312105	.24164
	% Difference		0.	2.4	8.1	15.7
.6	Exact Avail.			.31211	.24617	.20078
	Approx. Avail.			.31211	.24164	.18796
	% Difference			0.	1.8	6.4
.8	Exact Avail.				.18796	.14937
	Approx. Avail.				.18796	.14723
	% Difference				0.	1.4
1.0	Exact Avail.					.11628
	Approx. Avail.					.11628
	% Difference					0.

Number of Customers of Type 1 = 3
Number of Customers of Type 2 = 3
Number of Servers at Stage 1 = 4
Number of Servers at Stage 2 = 2
Number of Spares = 2

ρ_1	ρ_2	.2	.4	.6	.8	1.0
.2	Exact Avail.	.92381	.84612	.77751	.71959	.67105
	Approx. Avail	.92381	.82689	.70835	.58708	.47617
	% Difference	0.	2.3	8.9	18.4	29.0
.4	Exact Avail.	.80975	.66743	.56215	.48469	.42665
	Approx. Avail.	.78878	.66743	.54853	.44271	.35404
	% Difference	2.6	0.	2.4	8.7	16.9
.6	Exact Avail.	.70007	.52481	.411196	.33528	.28229
	Approx. Avail.	.62685	.51148	.411196	.32834	.26191
	% Difference	10.5	2.5	0.	2.1	7.2
.8	Exact Avail.	.60657	.42019	.31102	.24301	.19792
	Approx. Avail.	.47617	.38165	.30448	.24301	.19465
	% Difference	21.5	9.2	2.1	0.	1.7
1.0	Exact Avail.	.52984	.34405	.24348	.18403	.14607
	Approx. Avail.	.35404	.28237	.22557	.18098	.14607
	% Difference	33.2	17.9	7.4	1.7	0.

Number of Customers of Type 1 = 4
Number of Customers of Type 2 = 3
Number of Servers at Stage 1 = 4
Number of Servers at Stage 2 = 2
Number of Spares = 3

ρ_1	ρ_2	.2	.4	.6	.8	1.0
.2	Exact Avail.	.47783	.33118	.2448	.18867	.14995
	Approx. Avail.	.47783	.32026	.2200	.15455	.11083
	% Difference	0.	3.3	10.1	18.1	26.1
.4	Exact Avail.	.37758	.24871	.17722	.13289	.10339
	Approx. Avail.	.36489	.24871	.17345	.12355	.08976
	% Difference	3.4	0.	2.1	7.0	13.2
.6	Exact Avail.	.31467	.19940	.13803	.10119	.07734
	Approx. Avail.	.28185	.19511	.13803	.09964	.07329
	% Difference	10.4	2.1	0	1.5	5.2
.8	Exact Avail.	.27089	.16647	.11258	.08103	.06102
	Approx. Avail.	.22001	.15455	.11083	.08103	.06031
	% Difference	18.8	7.2	1.5	0.	1.2
1.0	Exact Avail.	.23846	.14293	.09482	.06721	.05000
	Approx. Avail.	.17345	.12355	.08976	.06643	.05000
	% Difference	27.3	13.6	5.3	1.2	0.

Number of Customers of Type 1 = 2
Number of Customers of Type 2 = 3
Number of Servers at Stage 1 = 4
Number of Servers at Stage 2 = 2
Number of Spares = 1

ρ_1	ρ_2	.2	.4	.6	.8	1.0
.2	Exact Avail.	.96431	.91565	.86711	.82255	.78279
	Approx. Avail.	.96431	.90388	.82037	.72557	.63006
	% Difference	0.	1.3	5.4	11.8	19.5
.4	Exact Avail.		.82037	.73618	.66642	.60934
	Approx. Avail.		.82037	.72557	.63006	.54088
	% Difference		0.	1.4	5.5	11.2
.6	Exact Avail.			.63006	.54810	.48481
	Approx. Avail.			.63006	.54088	.46154
	% Difference			0.	1.3	4.8
.8	Exact Avail.				.46154	.39748
	Approx. Avail.				.46154	.39305
	% Difference				0.	1.1
1.0	Exact Avail.					.33498
	Approx. Avail.					.33498
	% Difference					0.

Number of Customers of Type 1 = 3
Number of Customers of Type 2 = 3
Number of Servers at Stage 1 = 3
Number of Servers at Stage 2 = 2
Number of Spares = 3

ρ_1 \ ρ_2		.2	.4	.6	.8	1.0
.2	Exact Avail.	.39271	.26483	.19173	.14543	.11413
	Approx. Avail.	.39271	.25350	.16609	.11054	.07481
	% Difference	0.	4.3	13.4	23.99	34.4
.4	Exact Avail.		.166091	.11385	.08279	.06287
	Approx. Avail.		.166091	.11054	.07481	.05153
	% Difference		0.	2.9	9.6	18.0
.6	Exact Avail.			.07481	.05265	.03896
	Approx. Avail.			.07481	.05153	.03612
	% Difference			0.	2.1	7.3
.8	Exact Avail.				.036125	.026195
	Approx. Avail.				.036125	.025770
	% Difference				0.	1.6
1.0	Exact Avail.					.01869
	Approx. Avail.					.01869
	% Difference					0.

Number of Customers of Type 1 = 3
Number of Customers of Type 2 = 3
Number of Servers at Stage 1 = 5
Number of Servers at Stage 2 = 2
Number of Spares = 1

ρ_1 \ ρ_2		.2	.4	.6	.8	1.0
.2	Exact Avail.	.8208	.70712	.61945	.55047	.49514
	Approx. Avail.	.8208	.69106	.57184	.46957	.38473
	% Difference	0.	2.3	7.7	14.7	22.3
.4	Exact Avail.		.57184	.47756	.40846	.35597
	Approx. Avail.		.57184	.46957	.38473	.31553
	% Difference		0.	1.7	5.8	11.4
.6	Exact Avail.			.38473	.31969	.27201
	Approx. Avail.			.38473	.31553	.25953
	% Difference			0.	1.3	4.6
.8	Exact Avail.				.25953	.21659
	Approx. Avail.				.25953	.21434
	% Difference				0.	1.0
1.0	Exact Avail.					.17789
	Approx. Avail.					.17789
	% Difference					0.

Number of Customers of Type 1 = 3
Number of Customers of Type 2 = 3
Number of Servers at Stage 1 = 4
Number of Servers at Stage 2 = 3
Number of Spares = 2

ρ_1 \ ρ_2		.2	.4	.6	.8	1.0
.2	Exact Avail.	.59834	.40150	.29198	.22562	.18216
	Approx. Avail.	.59834	.36994	.22274	.13636	.08622
	% Difference	0.	7.9	23.7	39.6	52.7
.4	Exact Avail.		.222743	.143695	.10231	.07783
	Approx. Avail.		.222743	.136364	.08622	.05647
	% Difference		0.	5.1	15.7	27.4
.6	Exact Avail.			.086219	.058424	.04285
	Approx. Avail.			.086219	.056473	.03825
	% Difference			0.	3.3	10.7
.8	Exact Avail.				.038253	.027339
	Approx. Avail.				.038253	.026709
	% Difference				0.	2.3
1.0	Exact Avail.					.01916
	Approx. Avail.					.01916
	% Difference					0.

Number of Customers of Type 1 = 3
Number of Customers of Type 2 = 3
Number of Servers at Stage 1 = 4
Number of Servers at Stage 2 = 1
Number of Spares = 2

ρ_1 \ ρ_2		.2	.4	.6	.8	1.0
.2	Exact Avail.	.80461	.62529	.48700	.38456	.30873
	Approx. Avail.	.80461	.60772	.43897	.31211	.22208
	% Difference	0.	2.8	9.9	18.8	28.1
.4	Exact Avail.	.72270	.51828	.37852	.28424	.21942
	Approx. Avail.	.70489	.51828	.37041	.26306	.187959
	% Difference	2.5	0.	2.1	7.5	14.3
.6	Exact Avail.	.66305	.44811	.31211	.22580	.16927
	Approx. Avail.	.60772	.43897	.31211	.22208	.15958
	% Difference	8.3	2.0	0.	1.6	5.7
.8	Exact Avail.	.61702	.39829	.267397	.18796	.13772
	Approx. Avail.	.51828	.37041	.26306	.18796	.13596
	% Difference	16.0	7.0	1.6	0.	1.3
1.0	Exact Avail.	.58016	.36104	.23535	.16164	.11628
	Approx. Avail.	.43897	.31211	.22208	.15958	.11628
	% Difference	24.3	13.6	5.6	1.3	0.

Number of Customers of Type 1 = 2
Number of Customers of Type 2 = 4
Number of Servers at Stage 1 = 4
Number of Servers at Stage 2 = 2
Number of Spares = 2

APPENDIX B

DATA FOR MISCELLANEOUS RUNS TO INVESTIGATE SPARES AVAILABILITY, FLOW RATE, AND GENERAL TIME PROBABILITIES

Definitions

c_1 = Number of servers at stage 1

c_2 = Number of servers at stage 2

μ_{11} = Service rate for type 1 at stage 1

μ_{12} = Service rate for type 1 at stage 2

μ_{21} = Service rate for type 2 at stage 1

μ_{22} = Service rate for type 2 at stage 2

MMAX1 = Number of type 1 customers in system

MMAX2 = Number of type 2 customers in system

$AVAIL_R$ = Spares availability using average rate approximation

$AVAIL_E$ = Spares availability using exact analysis

$AVAIL_T$ = Spares availability using average time approximation

ΓR = Flow rate using average rate approximation

ΓE = Flow rate using exact analysis

ΓT = Flow rate using average time approximation

Gen Time AR = General time availability using average rate approximation

Gen Time AE = General time availability using exact analysis

Gen Time AT = General time availability using average time approximation

Data	Case Number 1	2	3	4	5	6
c_1	4.	4.	4.	1.	3.	3.
c_2	2.	3.	1.	3.	1.	1.
μ_{11}	1.	1.	1.	2.	1.	2.
μ_{12}	5.	5.	5.	1.	1.	1.
μ_{21}	1.	1.	1.	.2	.2	.2
μ_{22}	1.	1.	1.	1.	1.	1.
MMAX1	3.	3.	3.	3.	3.	3.
MMAX2	3.	3.	3.	3.	3.	3.
$AVAIL_R$	.6077	.6496	.3124	.9902	.2618	.0699
$AVAIL_E$	.4001	.4497	.1402	.9991	.4749	.3347
$AVAIL_T$	.3121	.3847	.0862	.9999	.6136	.5592
ΓR	3.634	3.799	2.726	1.096	.9765	.9986
ΓE	2.988	3.514	1.683	.3760	.8935	.9354
ΓT	2.853	3.325	1.649	.3636	.8302	.8649
Gen Time AR	.5521	.6170	.2129	.9866	.1420	.0212
Gen Time AE	.2989	.3951	.0590	.9993	.3542	.2217
Gen Time AT	.2226	.3199	.0355	.9999	.5094	.4434

Data	Case Number 7	8	9	10	11	12
c_1	3.	2.	3.	2.	3.	2.
c_2	2.	2.	2.	2.	3.	3.
μ_{11}	1.	1.	2.	2.	1.	1.
μ_{12}	1.	1.	1.	1.	1.	1.
μ_{21}	.2	.2	.2	.2	.2	.2
μ_{22}	1.	1.	1.	1.	1.	1.
MMAX1	3.	3.	3.	3.	3.	3.
MMAX2	3.	3.	3.	3.	3.	3.
$AVAIL_R$	.6301	.9087	.2862	.6460	.7545	.9587
$AVAIL_E$	.7828	.9661	.6426	.9255	.8530	.9834
$AVAIL_T$	.8780	.9855	.8527	.9806	.9233	.9937
ΓR	1.537	1.161	1.910	1.741	1.686	1.189
ΓE	1.173	.7354	1.385	.8447	1.229	.7439
ΓT	.9739	.6648	1.053	.7243	.9897	.6662
Gen Time AR	.5381	.8790	.1657	.5111	.7066	.9495
Gen Time AE	.7455	.9631	.5879	.9225	.8417	.9837
Gen Time AT	.8551	.9828	.8230	.9766	.9138	.9930

Data	Case Number 13	14	15	16	17	18
c_1	1.	3.	2.	1.	5.	1.
c_2	3.	3.	3.	6.	1.	1.
μ_{11}	1.	2.	2.	1.	1.	2.
μ_{12}	1.	1.	1.	1.	1.	4.
μ_{21}	.2	.2	.2	.2	.2	3.
μ_{22}	1.	1.	1.	1.	1.	4.
MMAX1	3.	3.	3.	3.	3.	1.
MMAX2	3.	3.	3.	3.	3.	1.
$AVAIL_R$	.9992	.4483	.8061	.99964	.0204	.6154
$AVAIL_E$	.9998	.7359	.9576	.99989	.0453	.6190
$AVAIL_T$	.9999	.9071	.9916	.99998	.1101	.6250
ΓR	.5999	2.480	2.018	.6000	.9932	2.016
ΓE	.3388	1.533	.8703	.3389	.9836	1.954
ΓT	.333	1.076	.7265	.3333	.9381	1.959
Gen Time AR	.9991	.3369	.7394	.99961	.0068	.4961
Gen Time AE	.9998	.7223	.9620	.99991	.0164	.5116
Gen Time AT	.9999	.8946	.9905	.99997	.0619	.5102

APPENDIX C

PROOF THAT $AVAIL_R \leq AVAIL_E \leq AVAIL_T$ FOR SINGLE SERVERS, SINGLE CUSTOMER OF EACH TYPE

This appendix proves analytically that the exact system availability, $AVAIL_E$, is bounded below and above by the approximations made by using an average rate and average time, $AVAIL_R$ and $AVAIL_T$, respectively, for a two-stage system with single servers at each stage and with one customer of each type in the system. Each customer has a different service rate at stage 1 and the same at stage 2.

Gross and Kahn (1976) proved the first half of this inequality for the average rate approximation. That will be repeated here and the proof for the other half of the inequality also given.

From page 5, Equation (3), of Gross and Kahn (1976) we have

$$AVAIL_E = \frac{\mu(\lambda_1 + \lambda_2 + 2\mu)}{2(\mu^2 + \lambda_1\mu + \lambda_2\mu + \lambda_1\lambda_2)}, \qquad \text{(C1)}$$

where

$AVAIL_E$ = exact spares availability

μ = service rate at stage 2 for both classes of customers

λ_i = service rate at stage 1 for customer of class i; i=1,2 .

The approximations involve substituting one rate for λ_1 and λ_2. Let it be $\bar{\lambda}$. Then the approximate spares availability expression (C1) becomes

$$APPROX\text{-}AVAIL = \frac{\mu(2\bar{\lambda} + 2\mu)}{2(\mu^2 + 2\bar{\lambda}\mu + \bar{\lambda}^2)}$$

$$= \frac{\mu}{\bar{\lambda} + \mu} . \qquad \text{(C2)}$$

We want to compare $AVAIL_E$ with the approximations. Thus, considering (C1) and (C2),

$$\frac{\mu(\lambda_1 + \lambda_2 + 2\mu)}{2(\mu^2 + \lambda_1\mu + \lambda_2\mu + \lambda_1\lambda_2)} \quad \text{versus} \quad \frac{\mu}{\bar{\lambda} + \mu}, \tag{C3}$$

or

$$\lambda_1\bar{\lambda} + \lambda_2\bar{\lambda} + 2\mu\bar{\lambda} - \lambda_1\mu - \lambda_2\mu - 2\lambda_1\lambda_2 \quad \text{versus} \quad 0 . \tag{C4}$$

For the average rate approximation ($AVAIL_R$), we have

$$\bar{\lambda} = \frac{\lambda_1 + \lambda_2}{2} . \tag{C5}$$

Substituting (C5) into (C4) and simplifying,

$$\lambda_1^2 - 2\lambda_1\lambda_2 + \lambda_2 \quad \text{versus} \quad 0$$

or

$$(\lambda_1 - \lambda_2)^2 \quad \text{versus} \quad 0 .$$

Since we have a squared term on the left, the inequality becomes $(\lambda_1 - \lambda_2)^2 \geq 0$, and thus $AVAIL_E \geq AVAIL_R$ from the comparison of (C3) and the direction in which the inequality has just been proven.

For the average time approximation ($AVAIL_T$), we have

$$\bar{\lambda} = \left[\frac{\frac{1}{\lambda_1} + \frac{1}{\lambda_2}}{2}\right]^{-1} = \frac{2\lambda_1\lambda_2}{\lambda_1 + \lambda_2} . \tag{C6}$$

Substituting (C6) into (C4) and simplifying,

$$2\mu\lambda_1\lambda_2 - \mu\lambda_1^2 - \mu\lambda_2^2 \quad \text{versus} \quad 0$$

$$-\mu(\lambda_1 - \lambda_2)^2 \quad \text{versus} \quad 0 .$$

Now the inequality is toward the left:

$$-\mu(\lambda_1 - \lambda_2)^2 \leq 0$$

and

$$AVAIL_E \leq AVAIL_T ,$$

by the comparison of (C3) and the direction in which the inequality has just been proven.

This completes the proof.

THE GEORGE WASHINGTON UNIVERSITY
Program in Logistics
Distribution List for Technical Papers

The George Washington University
- Office of Sponsored Research
- Library
- Vice President H. F. Bright
- Dean Harold Liebowitz
- Mr. J. Frank Doubleday

ONR
- Chief of Naval Research
  - (Codes 200, 430D, 1021P)
  - Resident Representative

OPNAV
- OP-40
- DCNO, Logistics
- Navy Dept Library
- OP-911
- OP-964

Naval Aviation Integrated Log Support

NAVCOSSACT

Naval Cmd Sys Sup Activity Tech Library

Naval Electronics Lab Library

Naval Facilities Eng Cmd Tech Library

Naval Ordnance Station
- Louisville, Ky.
- Indian Head, Md.

Naval Ordnance Sys Cmd Library

Naval Research Branch Office
- Boston
- Chicago
- New York
- Pasadena
- San Francisco

Naval Research Lab
- Tech Info Div
- Library, Code 2029 (ONRL)

Naval Ship Engng Center
- Philadelphia, Pa.
- Hyattsville, Md.

Naval Ship Res & Dev Center

Naval Sea Systems Command
- Tech Library
- Code 073

Naval Supply Systems Command
- Library
- Capt W. T. Nash

Naval War College Library
- Newport

BUPERS Tech Library

FMSO

Integrated Sea Lift Study

USN Ammo Depot Earle

USN Postgrad School Monterey
- Library
- Dr. Jack R. Borsting
- Prof C. R. Jones

US Marine Corps
- Commandant
- Deputy Chief of Staff, R&D

Marine Corps School Quantico
- Landing Force Dev Ctr
- Logistics Officer

Armed Forces Industrial College

Armed Forces Staff College

Army War College Library
- Carlisle Barracks

Army Cmd & Gen Staff College

US Army HQ
- LTC George L. Slyman
- Army Trans Mat Command

Army Logistics Mgmt Center
- Fort Lee

Commanding Officer, USALDSRA
- New Cumberland Army Depot

US Army Inventory Res Ofc
- Philadelphia

HQ, US Air Force
- AFADS-3

Griffiss Air Force Base
- Reliability Analysis Center

Maxwell Air Force Base Library

Wright-Patterson Air Force Base
- HQ, AF Log Command
- Research Sch Log

Defense Documentation Center

National Academy of Science
- Maritime Transportation Res Board Library

National Bureau of Standards
- Dr E. W. Cannon
- Dr Joan Rosenblatt

National Science Foundation

National Security Agency

WSEG

British Navy Staff

Logistics, OR Analysis Establishment
- National Defense Hdqtrs, Ottawa

American Power Jet Co
- George Chernowitz

ARCON Corp

General Dynamics, Pomona

General Research Corp
- Dr Hugh Cole
- Library

Planning Research Corp
- Los Angeles

Rand Corporation
- Library

Carnegie-Mellon University
- Dean H. A. Simon
- Prof G. Thompson

Case Western Reserve University
- Prof B. V. Dean
- Prof John R. Isbell
- Prof M. Mesarovic
- Prof S. Zacks

Cornell University
- Prof R. E. Bechhofer
- Prof R. W. Conway
- Prof J. Kiefer
- Prof Andrew Schultz, Jr.

Cowles Foundation for Research
- Library
- Prof Herbert Scarf
- Prof Martin Shubik

Florida State University
- Prof R. A. Bradley

Harvard University
- Prof K. J. Arrow
- Prof W. G. Cochran
- Prof Arthur Schleifer, Jr.

New York University
- Prof O. Morgenstern

Princeton University
- Prof A. W. Tucker
- Prof J. W. Tukey
- Prof Geoffrey S. Watson

Purdue University
Prof S. S. Gupta
Prof H. Rubin
Prof Andrew Whinston

Stanford
Prof T. W. Anderson
Prof G. B. Dantzig
Prof F. S. Hillier
Prof D. L. Iglehart
Prof Samuel Karlin
Prof G. J. Lieberman
Prof Herbert Solomon
Prof A. F. Veinott, Jr.

University of California, Berkeley
Prof R. E. Barlow
Prof D. Gale
Prof Rosedith Sitgreaves
Prof L. M. Tichvinsky

University of California, Los Angeles
Prof J. R. Jackson
Prof Jacob Marschak
Prof R. R. O'Neill
Numerical Analysis Res Librarian

University of North Carolina
Prof W. L. Smith
Prof M. R. Leadbetter

University of Pennsylvania
Prof Russell Ackoff
Prof Thomas L. Saaty

University of Texas
Prof A. Charnes

Yale University
Prof F. J. Anscombe
Prof I. R. Savage
Prof M. J. Sobel
Dept of Admin Sciences

Prof Z. W. Birnbaum
University of Washington

Prof B. H. Bissinger
The Pennsylvania State University

Prof Seth Bonder
University of Michigan

Prof G. E. P. Box
University of Wisconsin

Dr. Jerome Bracken
Institute for Defense Analyses

Prof H. Chernoff
MIT

Prof Arthur Cohen
Rutgers – The State University

Mr Wallace M. Cohen
US General Accounting Office

Prof C. Derman
Columbia University

Prof Paul S. Dwyer
Mackinaw City, Michigan

Prof Saul I. Gass
University of Maryland

Dr Donald P. Gaver
Carmel, California

Dr Murray A. Geisler
Logistics Mgmt Institute

Prof J. F. Hannan
Michigan State University

Prof H. O. Hartley
Texas A & M Foundation

Mr Gerald F. Hein
NASA, Lewis Research Center

Prof W. M. Hirsch
Courant Institute

Dr Alan J. Hoffman
IBM, Yorktown Heights

Dr Rudolf Husser
University of Bern, Switzerland

Prof J. H. K. Kao
Polytech Institute of New York

Prof W. Kruskal
University of Chicago

Prof C. E. Lemke
Rensselaer Polytech Institute

Prof Loynes
University of Sheffield, England

Prof Steven Nahmias
University of Pittsburgh

Prof D. B. Owen
Southern Methodist University

Prof E. Parzen
State University New York, Buffalo

Prof H. O. Posten
University of Connecticut

Prof R. Remage, Jr.
University of Delaware

Dr Fred Rigby
Texas Tech College

Mr David Rosenblatt
Washington, D. C.

Prof M. Rosenblatt
University of California, San Diego

Prof Alan J. Rowe
University of Southern California

Prof A. H. Rubenstein
Northwestern University

Dr M. E. Salveson
West Los Angeles

Prof Edward A. Silver
University of Waterloo, Canada

Prof R. M. Thrall
Rice University

Dr S. Vajda
University of Sussex, England

Prof T. M. Whitin
Wesleyan University

Prof Jacob Wolfowitz
University of Illinois

Mr Marshall K. Wood
National Planning Association

Prof Max A. Woodbury
Duke University